Universidad de Málaga

Escuela Técnica Superior de Ingeniería de Telecomunicación

TESIS DOCTORAL

Métodos y Herramientas para el Diseño y Verificación de Sistemas de Comunicaciones Móviles

Autor

Javier Poncela González

Director

José Tomás Entrambasaguas Muñoz

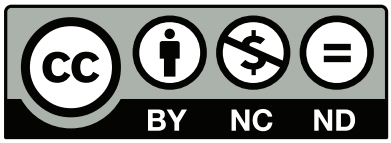
SPICUM servicio de publicaciones

UNIVERSIDAD DE MÁLAGA

SPICUM
servicio de publicaciones

AUTOR: Javier Poncela González

EDITA: Servicio de Publicaciones de la Universidad de Málaga



Esta obra está sujeta a una licencia Creative Commons:
Reconocimiento - No comercial - SinObraDerivada (cc-by-nc-nd):
Http://creativecommons.org/licences/by-nc-nd/3.0/es
Cualquier parte de esta obra se puede reproducir sin autorización
pero con el reconocimiento y atribución de los autores.
No se puede hacer uso comercial de la obra y no se puede alterar,
transformar o hacer obras derivadas.

Esta Tesis Doctoral está depositada en el Repositorio Institucional de
la Universidad de Málaga (RIUMA): riuma.uma.es

SPICUM
servicio de publicaciones

D. José Tomás Entrambasaguas Muñoz, profesor doctor del Departamento de Ingeniería de Comunicaciones de la Universidad de Málaga

CERTIFICA:

Que D. Javier Poncela González, Ingeniero de Telecomunicación, ha realizado en el Departamento de Ingeniería de Comunicaciones de la Universidad de Málaga, bajo su dirección el trabajo de investigación correspondiente a su TESIS DOCTORAL titulada:

"Métodos y Herramientas para el Diseño y Verificación de Sistemas de Comunicaciones Móviles"

En dicho trabajo se han propuesto aportaciones originales para el diseño de sistemas de pruebas para la certificación de equipos de comunicaciones móviles tanto en procedimientos de conformidad como de interoperatividad. Se ha propuesto una metodología de diseño de estos sistemas de pruebas que utiliza lenguajes de la familia ITU, y que está soportada por un entorno integrado de desarrollo formado por herramientas tanto comerciales como propias. La metodología está basada en una arquitectura genérica, propuesta en la Tesis, válida para todo sistema de pruebas. La metodología, herramientas y arquitectura han sido aplicadas también a sistemas de pruebas radio. Estas aportaciones han sido validadas con la implementación de sistemas de pruebas para las tecnologías DECT, Bluetooth y UMTS; los dos últimos han sido comercializados a nivel mundial.

Por todo ello, considera que esta Tesis es apta para su presentación al Tribunal que ha de juzgarla. Y para que conste a efectos de lo establecido en el Artículo 8º del Real Decreto 778/1998, regulador de los Estudios de Tercer Ciclo-Doctorado, AUTORIZA la presentación de esta Tesis en la Universidad de Málaga.

Málaga a ___6____ de _____Mayo____ de 2009

Fdo: José Tomás Entrambasaguas Muñoz

SPICUM servicio de publicaciones

SPICUM
servicio de publicaciones

UNIVERSIDAD DE MÁLAGA

ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA DE TELECOMUNICACIÓN

Reunido el tribunal examinador en el día de la fecha, constituido por:

Presidente:Dr. D. _____

Secretario: Dr. D. _____

Vocales: Dr. D. _____

Dr. D. _____

Dr. D. _____

para juzgar la Tesis Doctoral titulada **"Métodos y Herramientas para el Diseño y Verificación de Sistemas de Comunicaciones Móviles"**, presentada por D. Javier Poncela González y dirigida por Dr. D. J. Tomás Entrambasaguas Muñoz,

acordó por _____ otorgar la calificación de

_____

y, para que conste, se extiende firmada por los componentes del tribunal la presente diligencia.

Málaga, a ____ de _____ de _____

El presidente: El secretario:

Fdo: _____ Fdo: _____

El vocal: El vocal: El vocal:

Fdo: _____ Fdo: _____ Fdo: _____

SPICUM servicio de publicaciones

SPICUM
servicio de publicaciones

Aunque llegó más tarde, quiero
dedicar este trabajo a quien
es el motivo de que lo haya
terminado y que me hace
sonreír en las fotos.

SPICUM
servicio de publicaciones

SPICUM
servicio de publicaciones

AGRADECIMIENTOS

Esta Tesis ha sido un largo proyecto con el que han estado relacionadas muchas personas. A todas ellas quisiera agradecerles tanto su ayuda como las sugerencias que me han ofrecido. Sin embargo, hay algunas de estas personas a las que quisiera especialmente agradecer su participación.

En primer lugar, a Tomás, por su enfoque y sus comentarios. Como director de esta Tesis, ha sido el que más directamente ha seguido su progreso y tengo claro que sin su visión este trabajo no habría salido adelante.

A Rafa, Pablo y Ricardo por haber participado en los comienzos de esta Tesis, ayudando a demostrar algunas ideas y descartar otras. A Juan Pablo, por sus análisis y su capacidad de abstracción; aún me recuerdan alguna de sus respuestas al tribunal. A Lourdes, Bea y Victoria por su capacidad de trabajo, su confianza y su esfuerzo para llevar a la práctica indicaciones a menudo demasiado genérica; a Lourdes también agradecerle su ayuda en las últimas etapas de este trabajo. A Leopoldo, Isaac, Alejandro y Miguel Ángel, muchas gracias por la camisa. A Juan, Sergio, Carlos y Antonio, gracias por haber ayudado a mostrar que es posible aplicar conceptos previos de formas diversas. Vuestra colaboración ha sido inestimable.

También quiero mencionar a José Alberto, Casimiro, Nacho y Carlos, aunque su trabajo no se muestra en esta Tesis contribuyeron con sus desarrollos a demostrar el uso de Bluetooth en aplicaciones diversas. Igualmente, agradecer a Lidia, Mª José y Ana, a Juan Antonio, Raquel, Mª José y Elisa su dedicación. A Jesús, Jacobo, Antonio, Francisco, Antonio M., Fernando y Antonio J. por sus herramientas de edición, análisis y compilación. A Unai, Fernando, Alberto e Inés, por haber guiado a otros en un área que no es la suya cuando se lo pedí.

A Joaquín, que con sus dos gorras, ha llevado las ideas de este trabajo por todo el mundo. Y cómo no, a Carlos, Luis Fernando, Janie y Tomás, otra vez, que hicieron posible el primer proyecto de investigación a partir del cual surgió esta Tesis y de los cuales he recibido todo el apoyo posible a lo largo de todo este trabajo.

SPICUM servicio de publicaciones

SPICUM
servicio de publicaciones

RESUMEN

El objetivo de esta Tesis ha sido la consecución de tecnología para el diseño y mantenimiento eficientes, en coste temporal y económico, de Sistemas de Pruebas para equipos de comunicaciones inalámbricas. El proceso de diseño elaborado se ha basado en un conjunto de principios básicos: uso de lenguajes y notaciones ITU, independencia de la plataforma de ejecución y uso de herramientas comerciales.

En esta Tesis se describe:

- Una arquitectura genérica para Sistemas de Pruebas.
- Una Metodología de Diseño de Sistemas de Pruebas que establece las etapas a seguir y los documentos y modelos que cada etapa debe generar.
- Un conjunto de herramientas de soporte para la metodología, que, junto con herramientas comerciales, proporcionan un entorno de desarrollo que cubre las necesidades de todas las etapas.
- Una propuesta de metodología para el modelado con SDL de sistemas basados en estándares.
- La aplicación a Sistemas de Pruebas de capa física de la arquitectura, métodos y herramientas utilizados en el área de protocolos.
- Un conjunto de implementaciones que validan las aportaciones anteriores.

En la concepción inicial de este trabajo, el enfoque que se dio fue el de Sistemas de Pruebas orientados a la certificación de protocolos, es decir, dentro del marco de la Metodología de Pruebas de Conformidad OSI. Conforme se ha ido avanzando se ha visto que esta metodología es válida igualmente para otros enfoques distintos. Los resultados aplicables a las pruebas de conformidad son también aplicables a las pruebas de interoperatividad de protocolos. Además, se han aplicado los resultados obtenidos para el área de las pruebas de protocolos al ámbito de las pruebas de capa física, demostrando que es posible emplear la misma arquitectura, método y herramientas en ambos casos, lo que permite reducir los costes de desarrollo.

Para validar la Metodología de Diseño, se han construido Sistemas de Pruebas de conformidad de protocolos, sobre distintas plataformas, para los sistemas DECT, Bluetooth y UMTS. En los dos últimos casos el resultado han sido sendos Sistemas de Pruebas comerciales. La aplicación al área de las pruebas de capa física y al área de las pruebas de interoperatividad se ha demostrado sobre los sistemas Bluetooth y UMTS.

SPICUM servicio de publicaciones

SPICUM
servicio de publicaciones

ABSTRACT

The aim of this thesis has been the procurement of technology for efficient design and maintenance, both in terms of time and economic cost, of Test Systems for wireless communications equipment. The proposed design process has been based on a set of basic principles: use of ITU languages and notations, independence from the execution platform and use of commercial tools.

This thesis describes:

- A generic architecture for Test Systems.
- A Design Methodology for Test Systems which identifies the stages to be followed and the documents and models which each stage should generate.
- A set of support tools for the methodology which, together with commercial tools, provide a development environment that covers the needs of all stages.
- The application the architecture, methods and tools used in protocols to Test Systems for the physical layer.
- A proposed methodology for SDL modeling of systems based in standards.
- A set of implementations that validate the proposed methodology and the tools that have been implemented.

In the initial conception of this work, the approach taken was oriented towards Test Systems for the certification of protocols within the framework of the OSI Methodology for Conformance Testing. As work proceeded the methodology has been found equally valid for other approaches. The results applicable to conformance testing are also applicable to the interoperability testing of protocols. Furthermore, the results that have been obtained for the area of testing protocols have been also applied to the field of radio tests cases, showing that it is possible to use the same architecture, method and tools in both cases, thereby reducing development costs.

To validate the design methodology, conformance protocol Test Systems for DECT, Bluetooth and UMTS have been built on different platforms. In the latter two cases the resulting Test Systems have been commercialized. The application of the methodology in the areas of radio testing and interoperability testing has been demonstrated for Bluetooth and UMTS systems.

SPICUM servicio de publicaciones

SPICUM
servicio de publicaciones

CONTENIDOS

SPICUM servicio de publicaciones

SPICUM servicio de publicaciones

SPICUM servicio de publicaciones

SPICUM servicio de publicaciones

SPICUM servicio de publicaciones

SPICUM servicio de publicaciones

SPICUM servicio de publicaciones

SPICUM servicio de publicaciones

SPICUM servicio de publicaciones

SPICUM
servicio de publicaciones

LISTA DE FIGURAS

SPICUM servicio de publicaciones

SPICUM servicio de publicaciones

SPICUM servicio de publicaciones

SPICUM servicio de publicaciones

SPICUM servicio de publicaciones

SPICUM servicio de publicaciones

SPICUM servicio de publicaciones

SPICUM servicio de publicaciones

SPICUM servicio de publicaciones

SPICUM servicio de publicaciones

SPICUM servicio de publicaciones

SPICUM servicio de publicaciones

SPICUM servicio de publicaciones

SPICUM servicio de publicaciones

LISTA DE TABLAS

SPICUM servicio de publicaciones

SPICUM servicio de publicaciones

SPICUM servicio de publicaciones

SPICUM servicio de publicaciones

SPICUM servicio de publicaciones

SPICUM servicio de publicaciones

SPICUM servicio de publicaciones

SPICUM servicio de publicaciones

ACRÓNIMOS

A/D	*Analog-to-Digital*
ACL	*Asynchronous ConnectionLess*
ACLR	*Adjacent Channel Leakage Power Ratio*
ADPCM	*Adaptive Differential Pulse-Code Modulation*
AG	*Audio Gateway*
AM	*Acknowledged Mode*
ANSI	*American National Standards Institute*
ANTLR	*ANother Tool for Language Recognition*
API	*Application Programming Interface*
ARC	*Access Rights Class*
ARD	*Access Rights Details*
ARI	*Access Rights Identity*
AS	*Access Stratum*
ASCII	*American Standard Code for Information Interchange*
ASN.1	*Abstract Syntax Notation One*
ASP	*Abstract Service Primitive*
AT	*ATtention*
ATC	*Abstract Test Case*
ATM	*Abstract Test Method*
ATS	*Abstract Test Suite*
AuC	*Authentication Center*
AWT	*Abstract Windows Toolkit*
BB	*BaseBand*
BCCH	*Broadcast Control CHannel*
BCD	*Binary-Coded Decimal*
BCFE	*Broadcast Control Functional Entity*
BCH	*Broadcast Channel*
BER	*Basic Encoding Rules*
BI	*Behaviour Invalid*
BITE	*Bluetooth Qualification Tester*
BMC	*Broadcast Message Control* (DECT)
BMC	*Broadcast/Multicast Control* (UMTS)
BNF	*Backus-Naur Form*

SPICUM servicio de publicaciones

BV	*Behaviour Valid*
CA	*CApability*
CATG	*Computer Aided Test Generator*
CAT-iq	*Cordless Advanced Technology — internet and quality*
CBC	*Connectionless Bearer Control*
CC	*Call Control*
CCCH	*Common Control CHannel*
CCF	*Cluster Control Functions*
CCITT	*Comite Consultatif International de Telegraphique et Telephonique*
CHILL	*CCITT HIgh Level Language*
CI	*Common Interface*
CISS	*Call Independent Supplementary Services*
CLIP	*Calling Line Identification Presentation*
CLMS	*ConnectionLess Message Service*
CM	*Coordination Message*
CMC	*Connectionless Message Control*
CN	*Core Network*
COMS	*Connection Oriented Message Service*
CP	*Coordination Point*
CPCH	*Common Packet CHannel*
CRC	*Cyclic Redundancy Check*
CSF	*Cell Site Functions*
CSR	*Cambridge Silicon Radio*
CTCH	*Common Traffic CHannel*
CTFC	*Calculated Transport Format Combination*
CTM	*Cordless Terminal Mobility*
CTMF	*Conformance Testing Methodology and Framework*
CTP	*Cordless Telephony Profile*
CVI	*C Virtual Instrument*
D/A	*Digital-to-Analog*
DBC	*Dummy Bearer Control*
DC	*Dedicated Control*
DCCH	*Dedicated Control CHannel*
DCFE	*Dedicated Control Functional Entity*
DCH	*Dedicated CHannel*

SPICUM servicio de publicaciones

DECT	*Digital Enhanced Cordless Telecommunications*
DH	*Data-High rate*
DLC	*Data Link Connection* (Bluetooth)
DLC	*Data Link Control* (DECT)
DLCI	*Data Link Connection Identifier*
DLEI	*Data Link Endpoint Identifier*
DLI	*Data Link Identifier*
DM	*Data-Medium rate*
DMAP	*DECT Multimedia Access Profile*
DPCCH	*Dedicated Physical Control CHannel*
DPCH	*Dedicated Physical CHannel*
DPRS	*DECT Packet Radio Service*
DSCH	*Downlink Shared CHannel*
DSP	*Data Service Profiles*
DSP	*Digital Signal Processor*
DTCH	*Dedicated Traffic CHannel*
DTMF	*Dual Tone Multi-Frequency*
DTR	*Data Terminal Ready*
DV	*Data Voice*
EBDK	*Ericsson Bluetooth Development Kit*
EBNF	*Extended Backus-Naur Form*
ECN	*Exchanged Connection Number*
EDGE	*Enhanced Data for GSM Evolution*
EDR	*Enhanced Data Rate*
EFSM	*Extended Finite State Machine*
EG	*ETSI Guide*
EIR	*Equipment Identity Register*
EMC	*Equipment Manufacturer Code*
EN	*European Standard*
eODL	*extended Object Definition Language*
ES	*ETSI Standard*
ETR	*ETSI Technical Report*
ETS	*Executable Test Suite*
ETSI	*European Technical Standards Institute*
ETX	*End of TeXt*

SPICUM servicio de publicaciones

EUT	*Equipment Under Test*
EVM	*Error Vector Magnitude*
EWOS	*European Workshop for Open Systems*
FACH	*Forward Access Channel*
FDD	*Frequency Division Duplex*
FDT	*Formal Description Technique*
FEC	*Forward Error Correction*
FER	*Frame Error Rate*
FMCT	*Formal Methods on Conformance Testing*
FMID	*Fixed MAC Identity*
F-MMS	*Fixed line Multimedia Messaging Service*
FPGA	*Field-Programmable Gate Array*
FSM	*Finite State Machine*
FT	*Fixed Termination*
FTP	*File Transfer Profile*
GAP	*Generic Access Profile*
GC	*General Control*
GCF	*Global Certification Forum*
GCI	*Generic Compiler/Interpreter*
GenCod	*Generador de Codificador de la Interfaz*
GenCodecAir	*Generador de Codificador/Descodificador para la Interfaz Aire*
GenDef	*Generador de la Definición de la Interfaz*
GenInt	*Generador de Interfaces*
GERAN	*GSM/Edge Radio Access Network*
GFSK	*Gaussian Frequency Shift Keying*
GGSN	*Gateway GPRS Support Node*
GMSC	*Gateway Mobile Switching Center*
GOEP	*Generic Object Exchange Profile*
GPIB	*General Purpose Interface Bus*
GPRS	*General Packet Radio Service*
GSM	*Global System Mobile*
GSM-TS	*GSM Technical Specification*
GUI	*Graphical User Interface*
HCI	*Host Controller Interface*
HCTL	*Host Controller Transport Layer*

SPICUM servicio de publicaciones

HID	*Human Interface Devices*
HLR	*Home Location Register*
HMSC	*High-Level Message Sequence Chart*
HP	*Headset Profile*
HPVEE	*Hewlett-Packard Visual Engineering Environment*
HS	*Headset*
HSDPA	*High-Speed Downlink Packet Access*
HV	*High quality Voice*
I	*Information*
IAP	*ISDN Access Profile*
ICS	*Implementation Conformance Statement*
IEEE	*Institute of Electrical and Electronics Engineers*
IEI	*Information Element Identifier*
I-ETS	*Interim European Telecommunication Standard*
IMEI	*International Mobile Equipment Identity*
IMT-FT	*International Mobile Telecommunications – Frequency Time*
IP	*Internet Protocol*
IPEI	*International Portable Equipment Identity*
IPUI	*International Portable User Identity*
IPv6	*Internet Protocol version 6*
IRC	*Idle Receiver Control*
ISDN	*Integrated Services Digital Network*
ISM	*Industrial, Scientific & Medical*
ISO	*International Organization for Standardization*
ITEX	*Interactive TTCN Editor and Executor*
ITU	*International Telecommunications Union*
ITU-T	*International Telecommunication Union -Telecommunication Standardization Sector*
IUT	*Implementation Under Test*
IV	*Instrumento Virtual*
IWU	*InterWorking Unit*
IXIT	*Implementation eXtra Information for Testing*
JDK	*Java Development Kit*
JRE	*Java Runtime Environment*
L1	*Layer 1*

SPICUM servicio de publicaciones

L2	*Layer 2*
L2CAP	*Logical Link Control and Adaptation Protocol*
L3	*Layer 3*
LALR	*Look Ahead Left to Right*
LAN	*Local Area Network*
LANAP	*LAN Access Profile*
LAPC	*Link Access Procedure for Control plane*
LAPD	*Link Access Protocol on D channel*
LC	*Link Control*
LCE	*Link Control Entity*
LI	*Lengh indicator*
LL	*Left to right, Leftmost derivation*
LLAPI	*Low Level API*
LLME	*Lower Layer Management Entity*
LM	*Link Manager*
LMP	*Link Manager Protocol*
LR	*Left to right, Rightmost derivation*
LRMS	*Low Rate Message Service*
LSIG	*Link SIGnature*
LT	*Lower Tester*
LTE	*Long Term Evolution*
LUX	*Link U-plane service X*
LV	*Length-Value*
MAC	*Medium Access Control*
MB	*MegaByte*
MBC	*Multi-Bearer Control*
MCEI	*MAC Connection Endpoint Identification*
MDA	*Model-Driven Architecture*
MIB	*Master Information Block*
MINT	*Mobile communications INtegrated Tester*
MLAPI	*Medium Level API*
MM	*Mobility Management*
MOT	*Means Of Testing*
MP	*Machine Processable*
MS	*Mobile Station*

SPICUM servicio de publicaciones

MSC	*Message Sequence Chart*
MSC	*Mobile Switching Center*
M-SOMT	*Modified SOMT*
MTC	*Main Test Component*
MTS	*Methods for Testing & Specification*
NAS	*Non-Access Stratum*
NLF	*New Link Flag*
Nt	*Notification*
NWK	*NetWorK*
OBEX	*OBject eXchange Protocol*
OBW	*Occupied BandWidth*
ODAP	*Open Data Access Profile*
ODL	*Object Definition Language*
OMG	*Object Management Group*
OMT	*Object Modeling Technique*
OPP	*Object Push Profile*
OSI	*Open Systems Interconnection*
PABX	*Private Automatic Branch Exchange*
PACS	*Personal Access Communications System*
PAP	*Public Access Profile*
PARI	*Primary Access Rights Identity*
PARK	*Portable Access Right Key*
PC	*Personal Computer*
PCCH	*Paging Control CHannel*
PCCTS	*Purdue Compiler Construction Tool Set*
PCH	*Paging Channel*
PCI	*Peripheral Component Interconnect*
PCO	*Point of Control and Observation*
PCTR	*Protocol Conformance Test Report*
PDCP	*Packet Data Convergence Protocol*
PDU	*Protocol Data Unit*
PER	*Packed Encoding Rules*
PHL	*PHysical Layer*
PHS	*Personal Handy-phone System*
PHY	*PHYsical layer*

SPICUM servicio de publicaciones

PICS	*Protocol ICS*
PIM	*Platform-Independent Model*
PIXIT	*Protocol IXIT*
PMID	*Portable part MAC Identity*
PNFE	*Paging Notification Functional Entity*
PPP	*Point-to-Point Protocol*
PRECCX	*PRE-C-Compiler eXtended*
PSM	*Platform-Specific Model*
PSN	*Portable equipment Serial Number*
PSTS	*Profile Specific Test Specification*
PT	*Portable Termination*
PTC	*Parallel Test Component*
PTCC	*Protocol and Testing Competence Centre*
PTMSI	*Packet TMSI*
PTS	*Profile Test Specification*
PU	*Payload Unit*
PUN	*Portable User Number*
PUT	*Portable User Type*
RAB	*Radio Access Bearer*
RACH	*Random Access CHannel*
RATS	*Requirements Assistant for Telecommunications Services*
RB	*Radio Bearer*
RES	*Radio Equipment and Systems*
RF	*Radio Frequency*
RFCOMM	*Radio Frequency COMMunication*
RFP	*Radio Fixed Part*
RFPI	*Radio Fixed Part Identity*
RLC	*Radio Link Control*
RLL	*Radio in the Local Loop*
RNC	*Radio Network Controller*
RPN	*Radio fixed Part Number*
RR	*Receive Ready*
RRC	*Radio Resource Control*
RSSI	*Received Signal Strength Indication*
RTDX	*Real-Time Data Exchange*

SPICUM servicio de publicaciones

SAP	*Service Access Point*
SAPI	*Service Access Point Identifier*
SARI	*Secondary Access Rights Identity*
SCO	*Synchronous Connection Oriented*
SCPI	*Standard Commands for Programmable Instrumentation*
SCS	*System Conformance Statement*
SCTR	*System Conformance Test Report*
SDL	*Specification and Description Language*
SDL/GR	*SDL Graphic Representation*
SDL/PR	*SDL textual Phrase Representation*
SDP	*Service Discovery Protocol*
SDT	*SDL Design Tool*
SDU	*Service Data Unit*
SGSN	*Serving GPRS Support Node*
SHCCH	*SHared channel Control CHannel*
SIB	*System Information Block*
SIG	*Bluetooth Special Interest Group*
SIP	*Session Initiation Protocol*
SIR	*Signal-to-Interference Ratio*
SMS	*Short Message Service*
SOMT	*SDL-oriented Object Modeling Technique*
SPECS	*Specification and Programming Environment for Communication Software*
SPP	*Serial Port Profile*
SR	*Special Report*
SS	*System Simulator*
STX	*Start of TeXt*
SUT	*System Under Test*
TAC	*Test Adaptor Codec*
TAL	*Test Adaptor Logging*
TAM	*Test Adaptor Management*
TARI	*Tertiary Access Rights Identity*
TB	*Transport Block*
TBC	*Traffic Bearer Control*
TBR	*Technical Basis for Regulation*

SPICUM servicio de publicaciones

TC MTS	*Technical Committee Methods for Testing & Specification*
TCL	*Test Case Library*
TCP	*Transmission Control Protocol*
TCP	*Test Control Procedures*
TCRTR	*Technical Committee Reference Technical Report*
TCS	*Telephony Control protocol Specification*
TCTR	*Technical Committee Technical Report*
TDD	*Time Division Duplex*
TDMA	*Time Division Multiple Access*
TF	*Transport Format*
TFC	*Transport Format Combination*
TFCI	*Transport Format Combination Indicator*
TFCS	*Transport Format Combination Set*
TFI	*Transport Format Identifier*
TFS	*Transport Format Set*
TFSS	*Transport Format Set Set*
TM	*Transparent Mode*
TME	*Transfer Mode Entity*
TMP	*Test Management Protocol*
TMSI	*Temporary Mobile Subscriber Identity*
T-MUX	*Tail MUltipleXer*
TPUI	*Temporary Portable User Identity*
TR	*Technical Report*
TRUP	*TRansparent UnProtected service*
TS	*ETSI Technical Specification*
TS	*Test Step*
TSO	*Test Suite Operation*
TSS&TP	*Test Suite Structure & Test Purposes*
TTCN	*Testing and Test Control Notation*
TTCN	*Tree and Tabular Combined Notation*
TTCN.MP	*Tree and Tabular Combined Notation Machine Processable*
TTCN.PR	*Tree and Tabular Combined Notation Graphical Representation*
TTCN-RB	*TTCN Runtime Behaviour*
TTI	*Transmission Time Interval*
UART	*Universal Asynchronous Receiver-Transmitter*

SPICUM servicio de publicaciones

UE	*User Equipment*
UM	*Unacknowledged Mode*
UML	*Unified Modeling Language*
UMTS	*Universal Mobile Telecommunications System*
URN	*User Requirements Notation*
USB	*Universal Serial Bus*
UT	*Upper Tester*
UTRAN	*UMTS Terrestrial Radio Access Network*
VER	*Bit Error Rate*
VLR	*Visitor Location Register*
VXI	*VME eXtensions for Instrumentation*
WCDMA	*Wideband Code Division Multiple Access*
WiMAX	*Worldwide Interoperability for Microwave Access*
XML	*eXtensible Markup Language*
yacc	*Yet Another Compiler Compiler*

SPICUM servicio de publicaciones

SPICUM
servicio de publicaciones

INTRODUCCIÓN

SPICUM
servicio de publicaciones

SPICUM
servicio de publicaciones

Capítulo

1

"Tengo una historia que contarte. Mi propia implicación directa fue fugazmente breve y ni siquiera he pensado en presentarme yo mismo con algo tan presuntuoso como un nombre. No obstante, yo estaba allí, en el mismo principio de uno de esos principios."

Anónimo, El Algebrista

CAPÍTULO 1: INTRODUCCIÓN

Como usuarios de redes de telecomunicación nos hemos acostumbrado a poder acceder a los servicios que prestan con la misma familiaridad con que esperamos que una bombilla ilumine la estancia al pulsar un interruptor. En la sociedad ha ido calando el paradigma de la conectividad total, en todo momento, a todo el mundo, con cualquier servicio. En este complejo entramado de redes, una miríada de suministradores se afanan por situar sus productos ofreciendo funcionalidades cada vez más extensas en un mercado que se define principalmente por la novedad, lo visual y una relación coste-prestaciones continuamente decreciente.

Cuando un usuario adquiere un equipo de comunicación está aceptando un contrato con el suministrador en el que, a cambio de una contraprestación económica, se le entrega un dispositivo que le permite acceder a parte de este mundo incorpóreo de datos, con el entendimiento implícito de que el equipo no sólo es capaz de realizar su función, sino que también cumple las reglamentaciones en materia de telecomunicación. Para garantizar la veracidad de estas afirmaciones, el suministrador debe realizar distintas comprobaciones a lo largo del proceso de diseño (Figura 1.1) con objeto de verificar el mismo, pero, además, la puesta a la venta del producto requiere, según la legislación vigente, un certificado oficial. Este certificado sirve como garantía de que el equipo es capaz de comunicarse correctamente sin perturbar la operación de otros sistemas, para lo cual debe someterse a diversas comprobaciones estáticas y dinámicas.

1.1 Concepto de Prueba

El término probar deriva del latín *probare*. Según el Diccionario de la Real Academia de la Lengua [DRAE01], probar significa "*hacer examen y experimento de las cualidades de personas o cosas*". Una prueba sería la "*razón, argumento, instrumento u otro medio con que se pretende mostrar y hacer patente la verdad o falsedad de algo*". Según Hetzel, una prueba es "*cualquier actividad orientada a evaluar un atributo o capacidad*

SPICUM servicio de publicaciones

de un programa o sistema y determinar que cumple los requisitos establecidos" [HETZ88].

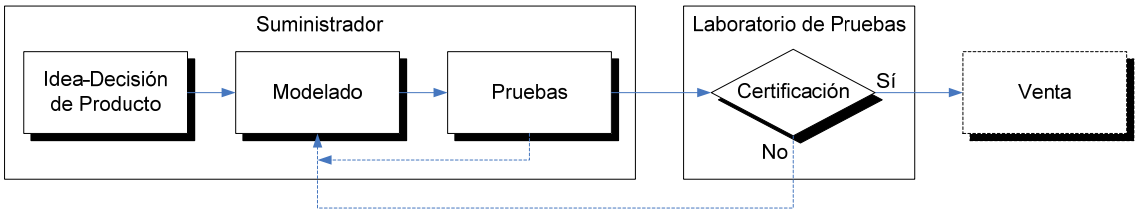


Figura 1.1: Etapas en el diseño de un sistema comercial de comunicación.

Al hablar de pruebas la primera clasificación que surge es la división entre pruebas de caja blanca y pruebas de caja negra. La diferencia entre una y otra radica en la posibilidad de acceder (caja blanca) o no (caja negra) a todos los elementos de la implementación. En el caso del software, al código fuente; si se trata de hardware, a sus esquemas eléctricos.

La variedad de técnicas de prueba existentes es muy amplia. Dentro de las pruebas de caja blanca se pueden citar la inspección de código, el análisis de flujo o las pruebas de condiciones, pudiendo ir asociadas a una medida del grado de cobertura alcanzado en la implementación ([BEIZ90], [GRAH94]). Algunas de las técnicas más utilizadas para el caso de caja negra son las pruebas aleatorias, el análisis causa-efecto, las pruebas de máquinas de estado finito, las pruebas basadas en lógica, etc.

Evidentemente, cuando se trata de sistemas de comunicación las pruebas blancas suponen un riesgo obvio para el suministrador, al tener que franquear el acceso a la totalidad de su diseño, pudiéndose violar derechos de propiedad intelectual. Por ello, las pruebas utilizadas en el proceso de certificación son del tipo de caja negra.

1.2 Pruebas de Sistemas

Los equipos comerciales de comunicación se definen como sistemas abiertos donde no se impone una implementación concreta de la funcionalidad, sino que se fijan las interfaces externas y el comportamiento que deben mostrar. La especificación de un sistema de comunicación engloba dos tipos de documentos: las Especificaciones de Sistema, que describen la interacción de cada parte del sistema con las demás entidades, y las Especificaciones de Prueba, que definen las pruebas a las cuales se va a someter a los equipos para determinar su adecuación o no a las Especificaciones de Sistema. Las Especificaciones de Sistema están escritas en lenguaje natural, lo que hace posible que durante el proceso de diseño se introduzcan en el modelado errores no sólo debidos a un fallo en la implementación del comportamiento, sino también debidos a interpretaciones equivocadas a causa de ambigüedades en las especificaciones.

No es hasta finales de los años 80 cuando se define una metodología de prueba para sistemas abiertos universalmente aceptada, la Metodología de Pruebas de Conformidad de OSI (*Open Systems Interconnection*). Esta metodología formaliza todo el proceso de prueba desde la definición de las mismas hasta la provisión del certificado, incluyendo distintas arquitecturas de prueba y la notación en que deben modelarse las pruebas.

SPICUM servicio de publicaciones

Posteriormente, ITU (*International Telecommunications Union*) adoptó la metodología enfocándola a la certificación de sistemas de telecomunicación. Desde principios de los 90, con el sistema GSM (*Global System Mobile*), ETSI (*European Technical Standards Institute*), como organismo estandarizador a nivel europeo, normalizó, siguiendo esta metodología, el proceso de certificación para equipos que se vendieran en su ámbito de competencia. Dentro de ETSI, el encargado de liderar y promocionar la metodología de pruebas ha sido el comité MTS (*Methods for Testing & Specification*) ([MTS][1], [MTS96]). La Metodología de Pruebas de Conformidad ha sido aplicada con éxito a los sistemas de comunicaciones inalámbricas (GSM, DECT, Bluetooth, GPRS, UMTS, Hyperlan, WiMAX), así como a protocolos de la familia TCP/IP (IPv6, Mobile IP, SIP) [2].

Las pruebas de conformidad proporcionan una alta garantía de que un equipo vaya a funcionar adecuadamente tras su despliegue, sin perturbar el funcionamiento de los demás equipos presentes, ya pertenezcan al mismo sistema de comunicaciones o no. Este enfoque de pruebas está especialmente pensado para equipos que operen en redes que requieran autorización gubernamental para su instalación, como es el caso de las redes públicas de telefonía y datos fijas e inalámbricas, pero es válido para cualquier ámbito en el que se deseen realizar pruebas sobre equipos o sistemas de comunicación, abiertos o en propiedad.

Tabla 1.1: Número de páginas de las especificaciones de distintos sistemas.

	Especificaciones de Sistema	**Especificaciones de Prueba**
DECT	940	2000[3]
GSM	735	1530
Bluetooth	1520[4]	1610[3]
UMTS	1412	2216

Dentro de esta metodología, un Sistema de Pruebas es un equipo de referencia utilizado para verificar el comportamiento de una implementación. Para ello, emula el comportamiento del resto del sistema de comunicaciones, pudiendo provocar situaciones anómalas, y dispone de la capacidad de observar y medir los parámetros relevantes para las pruebas. Su complejidad es proporcional a la del propio sistema de comunicaciones, la cual ha ido creciendo (Tabla 1.1) con la definición de sistemas cada vez más avanzados, pero además incorpora funcionalidad adicional para poder crear escenarios de comportamiento anómalo.

Para los sistemas que no operan en un entorno público, el requisito de certificación no existe, por lo que las pruebas de conformidad no han tenido el mismo impacto. Sin embargo, desde hace una década ha ido cogiendo impulso la realización de pruebas de

[1] Este comité ha sustituido al original Centro de Competencia de Pruebas y Protocolos (PTCC – *Protocol and Testing Competence Centre*).

[2] IPv6 – *Internet Protocol version 6*; SIP – *Session Initiation Protocol*.

[3] En el Sistema DECT se prueban tanto la Terminación Portátil como la Terminación Fija, de ahí el hecho de que aparezca un número tan elevado de páginas.

[4] Las Especificaciones de Sistema de Bluetooth emplean un formato de documento que hace que ocupe más páginas, aunque la cantidad de información sea menor. Esto explica la diferencia en volumen de páginas con respecto al sistema GSM.

SPICUM servicio de publicaciones

interoperatividad, es decir, comprobar el funcionamiento de una implementación haciéndola operar con implementaciones de otros suministradores. Esto puede servir para identificar errores en la interpretación de las especificaciones en una fase temprana del diseño, los cuales pueden ser discutidos entre los suministradores logrando una interpretación consensuada que será la que se traslade a los productos comercializados. Se trata de un enfoque atractivo, válido en un entorno no regulado, que permite ahorrar costes sin una pérdida sustancial de las garantías ofrecidas al cliente. Las pruebas de interoperatividad, no obstante, han adoptado también las arquitecturas y la notación de prueba establecidas en la Metodología de Pruebas de Conformidad. Algunos ejemplos de sistemas donde se han utilizado con éxito son Bluetooth, IP y SIP.

1.3 Ingeniería de Sistemas

El término ingeniería se usa cuando los sistemas se diseñan y construyen usando principios científicos y matemáticos, junto con conocimientos aplicados sobre el dominio del problema. La ingeniería de sistemas se puede considerar como la elaboración sucesiva de una serie de modelos que representan, desde distintos puntos de vista, el sistema que se pretende construir. Un aspecto clave de esta visión es la elección del lenguaje adecuado para cada uno de estos modelos. De esta elección dependerá que el modelo cumpla o no su función, su inteligibilidad y el esfuerzo necesario para su elaboración. En último término, incluso la propia ingeniería depende de esta elección, que a su vez forma parte de ella. El proceso de ingeniería requiere la planificación de las distintas actividades. Se denomina planificación al proceso organizacional de crear y mantener un plan.

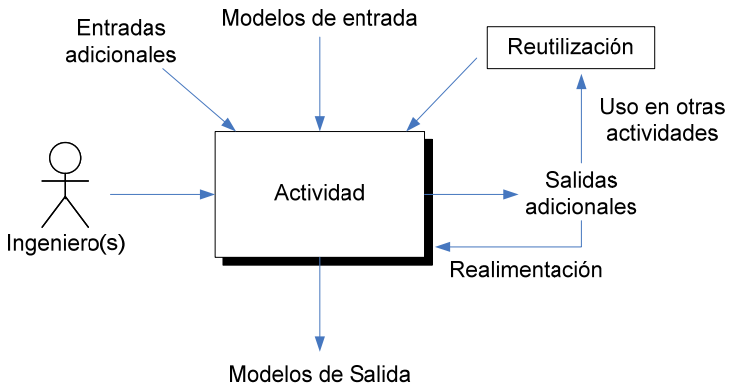


Figura 1.2: Esquema del proceso de ingeniería [OLSE94].

Un método individual es un modo sistemático de hacer algo. Una metodología es un conjunto integrado de métodos usado para un propósito particular. Los métodos contienen directrices, reglas e instrucciones que guían su ejecución. La diferencia entre una directriz y una regla es que una directriz da consejos que ayudan al proceso de ingeniería o mejoran la comprensión del sistema, mientras que una regla es un aserto que debe ser cumplido ya que de no hacerlo, el método podrá fallar.

A lo largo de esta memoria se utilizarán los términos sistema, modelo, arquitectura y plataforma según han definido OMG (*Object Management Group*) [MDA03] y [OLSE94]. Un sistema es una parte del mundo que es considerada como una unidad susceptible de ser modelada de forma independiente. Un sistema puede estar formado por una o más

SPICUM servicio de publicaciones

entidades, que pueden a su vez ser considerados como sistemas en otros ámbitos: una función, un programa, un equipo, una empresa, una combinación de partes de diferentes de sistemas, etc. Un modelo de un sistema es la descripción o especificación del sistema y su entorno con un cierto propósito. La arquitectura de un sistema es la especificación de las partes y conectores del sistema y las reglas para las interacciones entre las partes. Una plataforma es la agrupación de subsistemas y tecnologías que proporcionan un conjunto coherente de funcionalidad, a través de interfaces y patrones específicos de uso, que cualquier aplicación soportada puede utilizar sin preocuparse por los detalles de cómo se implementa.

1.4 Lenguajes de Diseño de Sistemas

Los sistemas de comunicaciones son vistos por ITU como máquinas de estados que interaccionan con su entorno y para ello ha definido una familia de lenguajes ([SG17], [ITUTZ], [ITUTX]) para modelar este comportamiento (SDL, MSC), las interfaces (ASN.1) y las secuencias de prueba (TTCN). La utilización de lenguajes formales ([WING94], [BRIN92]) constituye, como mínimo, una alternativa interesante a la hora de atacar la complejidad de estos sistemas, y aunque hay situaciones en que su uso no está justificado ([LUQI97], [HALL90]), en otros muchos casos ofrecen ventajas innegables sobre los lenguajes de programación general ([SIDH93], [DIET02]).

SDL (*Specification and Description Language*) ([Z.100], [ELLS97]) es un lenguaje concebido para modelar sistemas asíncronos, incorporando per se conceptos como señal, proceso, concurrencia o estado. Ofrece la ventaja de un diseño a alto nivel que muestra de forma gráfica tanto las relaciones entre elementos del sistema como el flujo del comportamiento del mismo. Fue creado en 1976 y ha ido evolucionando desde entonces, siendo sus versiones más importantes SDL'92 y SDL-2000. Los entornos de desarrollo comerciales para SDL ofrecen la posibilidad de simular los modelos del sistema, lo que permite su depuración en etapas tempranas del desarrollo, observando gráficamente el comportamiento. SDL ha sido adoptado internacionalmente por numerosos suministradores (Motorola, Nokia, …).

TTCN (*Tree and Tabular Combined Notation*) ([X.290] – [X.296]) es una notación específicamente pensada para la realización de pruebas. Ofrece construcciones básicas para el envío de estímulos y la recepción de respuestas, la comparación con patrones y la generación de veredictos. Surgido al tiempo que la Metodología de Pruebas de Conformidad OSI, ha evolucionado hasta una tercera versión, TTCN-3 (*Testing and Test Control Notation*) [ES 201 873][5], donde, aparte del formato tabular clásico, dispone de un formato gráfico similar a los diagramas MSC y un formato textual similar a lenguajes como Java. La definición de esta nueva versión del lenguaje ha estado abanderada por ETSI a través de su comité MTS (ver Apéndice B).

ASN.1 (*Abstract Syntax Notation One*) ([X.680] – [X.683]) es una notación formal que permite describir la sintaxis abstracta de la información intercambiada a través de la interfaz entre dos entidades. Esta notación proporciona distintos tipos básicos, a partir de los cuales es posible construir tipos más complejos, como estructuras y conjuntos. Al ser una notación abstracta no impone ninguna restricción sobre la forma de codificar y transmitir la información que modela, es decir, sobre la sintaxis de transferencia. Para

[5] El significado de las siglas TTCN fue cambiado en la versión TTCN-3.

SPICUM servicio de publicaciones

ello, y en asociación con ASN.1, se han definido conjuntos de reglas de codificación tales como BER [X.690] o PER [X.691].

MSC (*Message Sequence Chart*) [Z.120] es una notación que permite representar gráficamente la secuencia temporal de eventos que han tenido lugar en una o más entidades. También se conocen como diagramas de interacción y son similares a los diagramas de secuencia de UML (*Unified Modeling Language*) [UML07]. Permiten especificar el comportamiento externo en etapas tempranas del diseño, y son de ayuda a la hora de observar el comportamiento real de un sistema.

La familia de lenguajes ITU se completa con las notaciones URN y CHILL. URN (*User Requirements Notation*) [Z.150] permite capturar los requisitos de usuario, incluyendo requisitos no funcionales, en las etapas iniciales del diseño de un sistema, y mostrar gráficamente las dependencias entre ellos. CHILL (*CCITT*[6] *HIgh Level Language*) [Z.200] es un lenguaje de programación general definido para su utilización en sistemas de telecomunicación.

Aunque la definición de algunos de estos lenguajes comenzó hace ya treinta años, de forma implícita todos ellos asumen los principios que guían el relativamente reciente proceso de diseño software basado en la Arquitectura Dirigida por Modelos (MDA – *Model-Driven Architecture*) [MDA03]: separación entre el modelo y la implementación de un sistema. En MDA, la funcionalidad de un sistema se define como un modelo independiente de la plataforma (PIM – *Platform-Independent Model*), que posteriormente se traduce a un modelo específico de una plataforma concreta (PSM – *Platform-Specific Model*).

1.5 Objetivo de la Tesis

El objetivo de esta Tesis ha sido la consecución de tecnología para el diseño y mantenimiento eficientes, en coste temporal y económico, de Sistemas de Pruebas para equipos de comunicaciones inalámbricas. Un Sistema de Pruebas (Figura 1.3) se puede ver como un conjunto de funciones básicas, comunes a cualquier Sistema de Pruebas, más un grupo de funciones específicas. En la figura, los elementos en gris representan la funcionalidad de bajo nivel que se presupone incluida en la plataforma de ejecución o desarrollo, mientras que los elementos en verde representan funcionalidad realizada sobre la plataforma.

En esta Tesis se describe:

- Una arquitectura genérica para Sistemas de Pruebas.
- Una Metodología de Diseño de Sistemas de Pruebas que establece las etapas a seguir y los documentos y modelos que cada etapa debe generar.
- Un conjunto de herramientas de soporte para la metodología, que, junto con herramientas comerciales, proporcionan un entorno de desarrollo que cubre las necesidades de todas las etapas.
- La aplicación a Sistemas de Pruebas de capa física de la arquitectura, métodos y herramientas utilizados en el área de protocolos.

[6] *Comite Consultatif International de Télégraphique et Téléphonique.*

SPICUM servicio de publicaciones

- Una propuesta de metodología para el modelado con SDL de sistemas basados en estándares.
- Un conjunto de implementaciones que validan las aportaciones anteriores.

El proceso de diseño elaborado, que sigue la filosofía del diseño MDA, está basado en un conjunto de principios básicos, los cuales han servido como punto de partida: uso de lenguajes y notaciones ITU, independencia de la plataforma de ejecución, uso de herramientas comerciales y modularidad.

El diseño de un Sistema de Pruebas es todavía un arte, ya que se consideran, al igual que los equipos de comunicación, sistemas abiertos. El número potencial de ventas es muy bajo, en el entorno de las decenas de sistemas por lo que el coste de desarrollo tiene un impacto directo en la aceptación del producto. Pero este coste se puede reducir aumentando el nivel de abstracción y haciendo uso de herramientas de generación automática a partir de modelos de alto nivel.

La abstracción reduce los errores de diseño pero conlleva una pérdida de rendimiento. En los Sistemas de Pruebas, que tienen un alto coste final, se puede compensar esta pérdida de prestaciones con la utilización de plataformas más potentes. En conjunto, el diseño de alto nivel resulta en una ganancia neta que se hace patente tanto durante el desarrollo como durante el mantenimiento.

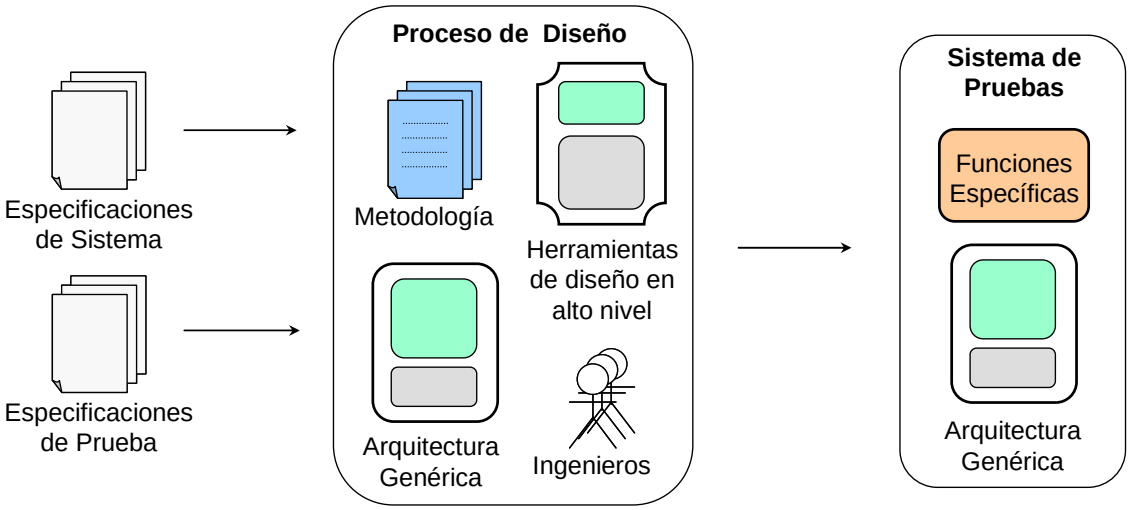


Figura 1.3: Visión abstracta del proceso de diseño de un Sistema de Pruebas.

Por ello, se ha utilizado SDL para el diseño de los protocolos contenidos en los Sistemas de Pruebas, mientras que se ha asumido que las pruebas se encuentran modeladas en TTCN. La interfaz entre modelos de ambas representaciones se describe mediante ASN.1. Los diagramas MSC se han empleado como elementos descriptivos auxiliares.

El diseño de sistemas en SDL no es sino un caso particular del problema más general del diseño de sistemas software. Se trata de un campo complejo, en el cual existe una gran variedad de alternativas. Por este motivo, se ha evaluado una metodología de diseño basado en SDL, SOMT (*SDL-oriented Object Modeling Technique*) [EKAN95], como posible guía. Al aplicar SOMT al diseño de sistemas basados en estándares, se pueden simplificar algunas de las actividades que sugiere. Como resultado se ha

SPICUM servicio de publicaciones

propuesto una metodología modificada (M-SOMT) para el diseño en SDL de sistemas basados en estándares.

Se ha buscado que la arquitectura de los Sistemas de Pruebas sea independiente de la plataforma de ejecución, facilitando la portabilidad y la selección de la mejor alternativa para cada situación. Los lenguajes de modelado SDL y TTCN facilitan esta tarea, ya que son abstractos, lo que hace que los modelos construidos sean, de por sí, independientes de la plataforma. La separación funcional de la arquitectura en bloques se ha realizado de manera que aquellos aspectos dependientes de la plataforma se encuentren agrupados. La adaptación de un Sistema de Pruebas a una nueva plataforma requiere únicamente la implementación de estos elementos. Se ha demostrado este punto ejecutando el Sistema de Pruebas para DECT en plataformas Linux, Solaris, Windows y DSP/BIOS.

Aunque la metodología es conceptualmente independiente de herramientas y notaciones concretas, y así se ha demostrado, su puesta en práctica requiere la realización de tareas que sólo son viables mediante el uso de herramientas automáticas. Estas herramientas incluyen editores, compiladores, generadores automáticos y simuladores. El desarrollo y mantenimiento de una herramienta propia supone un alto coste para cualquier compañía; por ello, para realizar las distintas tareas se han adoptado herramientas comerciales allí donde ha sido posible. Las carencias de las herramientas comerciales se han cubierto con el diseño de herramientas propias, fundamentalmente en dos áreas: componentes genéricos de la arquitectura y generadores automáticos de interfaces. Colateralmente, y aunque no son el objeto de esta tesis, se han desarrollado editores (SDL [GARC01], MSC [GALL00], TTCN [MART03], ODL - *Object Definition Language* [CABE00]), analizadores sintácticos y semánticos (SDL [COLA00], TTCN [LINA01]) y conversores (ASN.1-SDL [VALL00]).

La metodología ha sido aplicada al diseño de Sistemas de Pruebas, tanto de conformidad como de interoperatividad, para pruebas de protocolo y para pruebas de capa física. En primera instancia, fue validada mediante la construcción de un prototipo para equipos de comunicaciones DECT. Posteriormente, se ha utilizado para el diseño de Sistemas de Pruebas comerciales para equipos Bluetooth, UMTS y GERAN en colaboración con la empresa AT4 wireless [AT4W], que se ha convertido en un referente mundial en este área. Esta colaboración ha quedado reflejada en diversos proyectos de investigación ([PROY95], [PROY99], [PROY02], [PROY03], [CONT97], [CONT98], [CONT99a], [CONT99b], [CONT02], [CONT03a]).

La Metodología de Diseño ha ido evolucionando conforme así lo han hecho los sistemas de comunicación. Inicialmente, en el sistema DECT, se utilizaron únicamente los lenguajes SDL y TTCN. Aunque se intentó emplear ASN.1 para definir la interfaz entre ambos modelos, no fue posible dadas las limitaciones de las herramientas. Se desarrolló el conversor de tipos de datos ASN.1 a SDL, pero su mantenimiento se abandonó al incluir los entornos comerciales herramientas similares. Las pruebas de los sistemas Bluetooth y UMTS ya contienen declaraciones ASN.1, por lo que la metodología se modificó en este sentido. Igualmente, los generadores automáticos de interfaces fueron adaptados.

En la concepción inicial de este trabajo, el enfoque que se dio fue el de Sistemas de Pruebas orientados a la certificación de protocolos, es decir, dentro del marco de la Metodología de Pruebas de Conformidad OSI. Conforme se ha ido avanzando se ha visto que esta metodología es válida igualmente para otros enfoques distintos. Los resultados aplicables a las pruebas de conformidad son también aplicables a las pruebas

SPICUM servicio de publicaciones

de interoperatividad de protocolos. La única diferencia, como se ha comentado previamente, es que no se trata de una certificación regulatoria, sino simplemente de comprobaciones entre implementaciones de distintos suministradores. En este caso, es suficiente con se acuerde qué pruebas son las que se van a emplear.

Al igual que en el caso de los protocolos, todo equipo de comunicación que opere en un sistema regulado debe someterse a un proceso de certificación de su interfaz física. Una característica diferencial entre los Sistemas de Pruebas de protocolo y los Sistemas de Pruebas de capa física es el hecho de que los segundos requieren instrumentación específica capaz de llevar a cabo determinadas medidas sobre la interfaz eléctrica. Ejemplos de esta instrumentación son: generadores de señales, moduladores, osciloscopios, analizadores de espectros, etc. Esta instrumentación suele tener un coste bastante elevado, y, dado que habitualmente son necesarios varios equipos, ello provoca que los Sistemas de Pruebas de capa física sean considerablemente más caros que los Sistemas de Pruebas de protocolos.

La Metodología de Pruebas de Conformidad, según indica expresamente, no cubren plenamente el área de las pruebas de conformidad de capa física ([X.290]). El estándar TTCN ([X.292], [ES 201 873-1]) indica que su ámbito es el de la especificación de pruebas de sistemas reactivos pero también deja fuera la especificación de pruebas para protocolos de capa física. A pesar de esto, la certificación de la interfaz física de un sistema de comunicación se realiza siguiendo las mismas pautas que para los niveles de protocolos. La diferencia en el proceso radica en que las pruebas de protocolo se modelan en TTCN, mientras que las pruebas de capa física sólo se describen textualmente.

En esta tesis se han aplicado los resultados obtenidos para el área de las pruebas de protocolos al ámbito de las pruebas de capa física, demostrando que es posible emplear la misma arquitectura, método y herramientas en ambos casos, lo que permite reducir los costes de desarrollo.

1.6 Contenido de Capítulos

Esta memoria se ha organizado en doce capítulos, que corresponden a: a) una introducción, b) un capítulo de descripción de la Metodología de Pruebas, c) una sección de concepción más teórica, estructurada en cinco capítulos, donde se describe el trabajo realizado en el diseño de Sistemas de Pruebas, d) una sección de implementaciones, estructurada en cuatro capítulos, donde se demuestra la aplicación a sistemas comerciales de la metodología desarrollada, y e) unas conclusiones.

En el Capítulo 2 se resumen los conceptos relacionados con la Metodología de Pruebas de Conformidad. En primer lugar se distingue entre pruebas de conformidad y pruebas de interoperatividad. A continuación se explica qué es un Juego de Pruebas, visto como el conjunto de secuencias de interacción que verifican un conjunto de propósitos, y las posibles configuraciones de prueba, junto con los elementos que las forman. También se describen los documentos relacionados con el proceso de prueba y que son aplicables tanto a la realización de las pruebas como al diseño de un Sistema de Pruebas.

La **Sección I** cubre el aspecto teórico del Diseño de Sistemas de Pruebas. Se explican en esta sección la arquitectura genérica de pruebas, la Metodología de Diseño y las herramientas de soporte que se han desarrollado. Como complemento, se analiza una metodología de diseño software cuyas pautas pueden ser útiles para algunas actividades de la Metodología de Diseño.

SPICUM servicio de publicaciones

En el Capítulo 3 se presenta la arquitectura genérica de un Sistema de Pruebas. Esta arquitectura es modular y permite emplear distintas configuraciones a la hora de desplegar sus componentes sobre plataformas físicas. La arquitectura está compuesta por tres Subsistemas: Operación y Mantenimiento, de Pruebas e Inferior.

El Capítulo 4 describe la Metodología de Diseño propuesta. En primer lugar se ofrece una visión general de la misma, para, a continuación, describir la construcción de los Subsistemas de la arquitectura. La Metodología se ha estructurado en actividades, para cada una de las cuales se indican las entradas (modelos) que requiere y las salidas (modelos) que proporciona. A lo largo de la explicación se justifican las alternativas de diseño elegidas. Finalmente, se ofrece una posible planificación de los desarrollos basada en la experiencia adquirida.

En el Capítulo 5 se exponen las herramientas de soporte diseñadas como complemento de la Metodología de Diseño. En primer lugar se describen los componentes genéricos que se han desarrollado: Subsistema de Operación y Administración, Módulos de Adaptación a la Plataforma y Módulos de Gestión de las Pruebas. En un segundo bloque se describen los generadores automáticos de interfaces, utilizados tanto para la definición como para la codificación de la información que transportan.

El Capítulo 6 explica cómo es posible aplicar la Metodología de Diseño a los Sistemas de Pruebas de capa física. Se justifican las adaptaciones que sufren algunos de los componentes de la arquitectura para acomodarse a las características específicas de las pruebas de capa física. En concreto, se muestra cómo es posible modelar una medida física en TTCN, y cómo es posible utilizar instrumentación de diferentes fabricantes sin tener que modificar el resto de los elementos de la arquitectura.

El Capítulo 7 analiza la posibilidad de emplear la metodología SOMT para el diseño de parte del Sistema de Pruebas, en concreto, el Módulo de Protocolos. En el caso de estos sistemas, gran parte del análisis ha sido realizado previamente y se encuentra recogido en las Especificaciones de Sistema y las Especificaciones de Prueba, lo que sugiere simplificar o modificar algunas de las actividades. Así, se examina cada una de las actividades propuestas en SOMT bajo el prisma del diseño basado en estándares. Como resultado, se propone una metodología modificada (M-SOMT).

La **Sección II** describe los Sistemas de Pruebas que se han construido y que han permitido validar y refinar la Metodología de Diseño. En primer lugar se describe en detalle el diseño del prototipo de Sistema de Pruebas para DECT y, a continuación, se describen los aspectos novedosos de los diseños de Sistemas de Pruebas comerciales para Bluetooth y UMTS. Finalmente, se describen ejemplos de la aplicación de la Metodología de Diseño al diseño de Sistemas de Pruebas de capa física.

El Capítulo 8 explica detalladamente todas las tareas implicadas en el diseño de un Sistema de Prueba, los modelos resultantes de cada actividad y la utilización de las distintas herramientas, comerciales y propias. Una descripción del Sistema DECT se puede encontrar en el Apéndice F. Se presta un especial énfasis al diseño de los protocolos, por ser ésta la tarea más compleja y menos formal. Se describen también las pruebas que se han realizado a lo largo del diseño, y se muestra cómo adaptar el Sistema de Pruebas a distintas plataformas de ejecución.

En el Capítulo 9 se expone la aplicación de la Metodología de Diseño a Sistemas de Pruebas para Bluetooth empleando un entorno de desarrollo comercial diferente al empleado previamente. Se comprueba así que la Metodología es independiente de herramientas concretas, aunque requiere la adaptación de las interfaces a lo especificado

SPICUM servicio de publicaciones

en cada Subsistema. Se describen estas adaptaciones para los Sistemas de Pruebas de conformidad. Se presenta también el Sistema de Pruebas de interoperatividad desarrollado.

El Capítulo 10 cubre los Sistemas de Pruebas comerciales de conformidad e interoperatividad desarrollados para UMTS. En primer lugar se analizan las características de orientación a objetos de SDL. A continuación se exponen las modificaciones introducidas en los generadores automáticos por el hecho de utilizar ASN.1 en la definición de las pruebas. Finalmente, se describen los sistemas implementados.

El Capítulo 11 muestra ejemplos de aplicación de la Metodología de Diseño a Sistemas de Pruebas radio. Se presentan Sistemas de Pruebas radio para las tecnologías Bluetooth y UMTS siguiendo la arquitectura expuesta en el Capítulo 6, describiendo en detalle el modelado de una prueba radio de UMTS.

En las Conclusiones se recogen los principales resultados de esta Tesis y se proponen futuras líneas de trabajo. Los Apéndices A al L recogen información complementaria al contenido de los Capítulos.

1.6.1 Convenios

En la redacción de este documento se han utilizado los siguientes convenios.

Los términos que hacen referencia a elementos que son objeto de esta Tesis o parte de un estándar aparecen con la primera letra en mayúsculas ya que se asimilan a nombres propios. Cuando el mismo término se utiliza en su rol de término del lenguaje natural, el término se escribe en minúsculas. Ej: Sistema de Pruebas, Juego de Pruebas Abstractas, etc.

Parte de las figuras incluidas muestran distintos diseños en SDL. El Apéndice A resume el significado de los símbolos utilizados en la forma gráfica de este lenguaje.

Los términos que corresponden a código de programación se han escrito con formato de letra `Courier New`. Los nombres de las herramientas, así como los términos extranjeros, aparecen en *cursiva*.

SPICUM servicio de publicaciones

SPICUM
servicio de publicaciones

Capítulo

2

"Sus edificios eran sólidos, pero baratos: ensamblados a partir de combinaciones de hormigón de carbono, planchas de pasta de madera de resistencia uniforme, ladrillos hechos a partir de gránulos de arcilla cementada por bacterias genéticamente modificadas, vigas estructurales de acero esponjoso, cristal de sílice enlazado. A pesar de toda su estandarización, estos componentes básicos proporcionaban una abundancia de diversidad a los arquitectos."

Stephanie, El Dios Desnudo

CAPÍTULO 2: PRUEBAS DE CONFORMIDAD

En la década de los 70, al proliferar los sistemas distribuidos, los suministradores reconocieron la ventaja de los sistemas abiertos. El resultado fue la introducción de una serie de normas, la primera de las cuales tenía como objetivo definir la estructura global de todos los sistemas de comunicación. Esta norma fue introducida por ISO (*International Organization for Standardization*) y se conoce como el modelo de referencia para la interconexión de sistemas abiertos o modelo básico de referencia OSI [ISO 7498]. El enfoque adoptado por ISO para este modelo de referencia fue una organización por niveles, cada una de las cuales realiza una función bien definida. Cada nivel proporciona un conjunto de servicios al nivel inmediatamente superior y utiliza los servicios ofrecidos por el nivel inmediatamente inferior [HAL98].

El modelo de referencia, sin embargo, no indica cómo deben construirse estos niveles; tan sólo especifica los servicios que deben ofrecer. Por este motivo, cada suministrador puede tener su propia implementación. Las especificaciones, al estar escritas en lenguaje natural, son fuente de ambigüedades e interpretaciones erróneas. Podría ocurrir que dos implementaciones basadas en la misma especificación, se comportaran en la misma situación de forma diferente, constituyendo en la práctica dos versiones distintas de una misma norma. Como resultado, la interconexión de estas implementaciones plantearía serios problemas de comunicación al funcionar de facto como dos entidades diferentes.

La metodología OSI sobre pruebas de conformidad vino a cubrir el hueco originado por la posible existencia de interpretaciones diversas de una misma especificación o de fallos en las implementaciones. Esta metodología se encuentra recogida en el estándar [ISO 9646]. La idea es estandarizar el proceso de prueba de una implementación, lo que incluye tanto los mecanismos de prueba como la documentación necesaria durante el proceso. Como resultado se dispone de un procedimiento que permite verificar el

SPICUM servicio de publicaciones

comportamiento de las diferentes implementaciones. Se obtiene así tanto una mejora al formalizar el proceso de pruebas como una disminución del tiempo y número de pruebas necesario, pues evita la exponencialidad del proceso al aumentar la variedad de equipos.

A la hora de especificar las pruebas, la solución adoptada por ISO fue definir para cada protocolo o conjunto de ellos un conjunto de pruebas que se denomina Juego de Pruebas Abstractas (ATS – *Abstract Test Suite*). Al solicitar la certificación de un equipo, los suministradores deben declarar la funcionalidad de su implementación, denominada Implementación Bajo Prueba (IUT – *Implementation Under Test*)[1], en los correspondientes formularios. A partir de estas declaraciones, los laboratorios de pruebas se encargan de seleccionar el conjunto de pruebas aplicables a cada equipo y ejecutarlas. Como resultado generan un informe otorgando o no la certificación de conformidad según los veredictos obtenidos en las distintas pruebas.

A lo largo de este capítulo se desarrollan las ideas presentes en la norma ISO 9646 [ISO 9646] relevantes para el contenido de esta tesis. En la Sección 2.1 se presentan las ideas básicas de las pruebas, diferenciando entre las pruebas de conformidad y las pruebas de interoperatividad. La Sección 2.2 incluye una descripción de la Metodología de Pruebas de Conformidad propuesta por OSI y adoptada por la comunidad internacional.

2.1 Modalidades de Pruebas

Cuando se trata de comprobar una implementación existen, principalmente, dos alternativas: las pruebas de conformidad y las pruebas de interoperatividad. Estas alternativas se pueden ver como complementarias entre sí y los factores que influyen para seleccionar una de ellas son variados, pero se pueden citar los siguientes: si se trata de un producto que se va a vender para operar en un entorno con regulaciones estrictas, si se debe garantizar que su funcionamiento no interferirá en el de otros equipos, si el interés principal reside en su capacidad para dialogar con otros dispositivos, etc.

Las pruebas de conformidad buscan, fundamentalmente, certificar el funcionamiento de un equipo respecto a las especificaciones que lo regulan y determinar que no interfiera en la operación de los demás equipos [EWOS96]. Las pruebas de interoperatividad, por su parte, inciden en la capacidad de un equipo de interactuar con otros, relativizando un cumplimiento estricto de las especificaciones [LLOY89]. En esta sección se presentan ambas categorías de pruebas y sus características más reseñables.

Un aspecto importante de estos tipos de pruebas es que no tienen como finalidad el garantizar un funcionamiento correcto bajo toda circunstancia. La complejidad de los sistemas de comunicaciones hace que este objetivo sea, simplemente, inalcanzable, tanto por motivos de coste como de tiempo. Si consideramos un sistema de comunicaciones como una máquina de estados finitos (FSM – *Finite State Machine*; EFSM – *Extended Finite State Machine*), sería necesario construir un conjunto de pruebas que permitiera comprobar la implementación sometiéndola a todas las posibles combinaciones de señales de entrada y estados que pudieran darse a lo largo de su operación. La Figura 2.1 muestra un ejemplo donde aparece una máquina de estados finitos, con cuatro estados y cinco posibles señales de entrada, y algunas de las posibles

[1] En los estándares, el conjunto de todos los elementos necesarios para llevar a cabo la ejecución de las pruebas se denomina medio de pruebas (MOT – *Means Of Testing*), incluyendo tanto el equipo físico que realiza la prueba como todos los documentos y procedimientos definidos.

SPICUM servicio de publicaciones

secuencias válidas de entrada que pueden darse durante su operación. Incluso en un escenario tan simple, es fácil ver que el número de combinaciones es infinito.

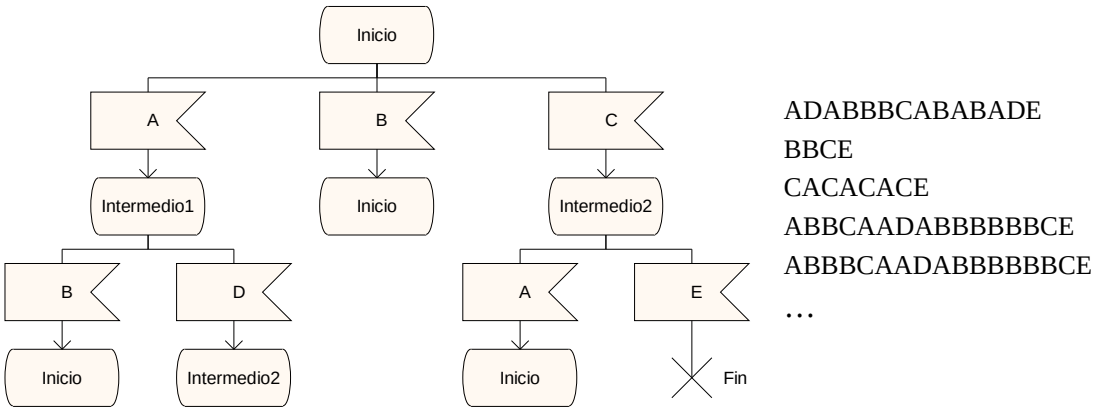


Figura 2.1: Ejemplo de máquina finita de estados y posibles secuencias válidas de entrada.

2.1.1 Pruebas de Conformidad

La Metodología de Pruebas de Conformidad OSI regula las pruebas realizadas por terceras partes. En esta situación raramente se encontrará disponible el código escrito; en realidad, pocos suministradores estarían dispuestos a poner a disposición de un tercero el código por ellos elaborado. Por tanto, las pruebas estarán habitualmente restringidas al caso de caja negra y su objetivo será la comprobación de que el comportamiento observable de la Implementación Bajo Prueba es correcto. La filosofía de caja negra de la metodología OSI queda reflejada en el enunciado de que "*sólo el comportamiento externo del sistema se recoge como el comportamiento estándar de los sistemas abiertos reales*" [ISO 7498].

Las pruebas de conformidad, tal y como se indica en la metodología, tratan de "*proporcionar una alta probabilidad de funcionamiento adecuado de la interfaz entre implementaciones de distintos suministradores*", lo cual incluye el reducir las posibilidades de que un equipo distorsione el funcionamiento de los demás equipos presentes en el sistema.

Se dice que un sistema es *conforme* a las especificaciones si cumple los requisitos de las especificaciones aplicables en su comunicación con otros sistemas [X.290]. Estos requisitos se dividen en requisitos estáticos y requisitos dinámicos:

- Requisitos de conformidad estática: Especifican limitaciones en las combinaciones de funcionalidad de la IUT. Se evalúan en función de una declaración del suministrador.
- Requisitos de conformidad dinámica: Especifican el comportamiento observable de la IUT.

A su vez, cualquier requisito debe pertenecer a una de estas tres categorías:

- Obligatorios: Deben cumplirse en toda circunstancia.
- Condicionales: Sólo es necesario que se cumplan si se dan ciertas condiciones.

SPICUM servicio de publicaciones

- Opcionales: Seleccionables de forma que se ajusten a la funcionalidad de la implementación.

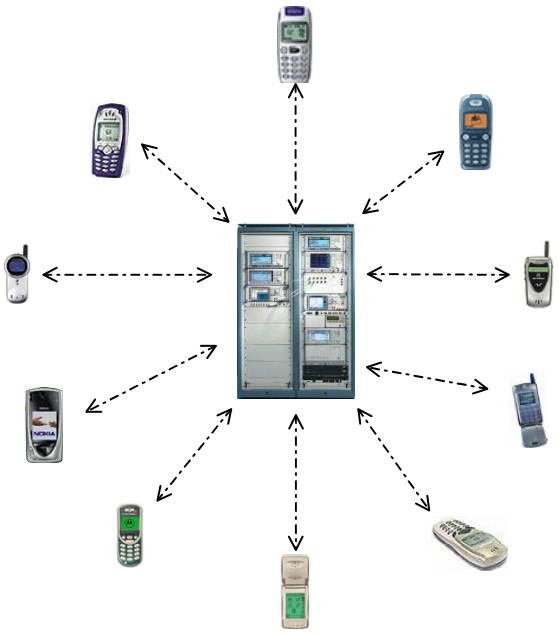


Figura 2.2: Esquema conceptual de las pruebas de conformidad.

La determinación de la conformidad o no de una IUT se realiza enfrentándola a un Sistema de Pruebas (Figura 2.2), sistema que ejecuta las pruebas necesarias para cada implementación. El resultado obtenido tras la ejecución de una prueba conlleva una cierta incertidumbre inevitable, pues no contempla todas las posibles variables externas, por ejemplo ambientales o de influencia de otros sistemas, que puedan afectar el comportamiento de la IUT. Algunos autores, en los estadios iniciales de la Metodología de Pruebas de Conformidad, indicaron que habían encontrado implementaciones plenamente conformes que encontraban problemas en su interacción [VERM94].

Si se obtiene un resultado positivo, servirá para incrementar la confianza en la capacidad de la IUT para interconectarse con otras implementaciones en la forma esperada[2]. Una prueba que genere un resultado de fallo revelará uno o más errores, ofreciendo alguna información concreta sobre la IUT; este resultado negativo obligará al suministrador a modificar su producto para cumplir las exigencias reglamentadas.

2.1.2 Pruebas de Interoperatividad

Las pruebas de interoperatividad se han propuesto como un complemento o, incluso, una alternativa de las pruebas de conformidad [BERT90]. Una posible definición del

[2] Esta idea es opuesta a la mayoría de pruebas de software donde se opina que un proceso de prueba que sólo genere un resultado de éxito ofrece poca información pues, aunque no hubo errores, se desconoce cuántos quedan sin descubrir. La distinta interpretación del resultado de la prueba se debe a la desconfianza de la corrección del software en general y a que las pruebas de conformidad se realizan al final del proceso de desarrollo, a lo largo del cual se habrán realizado pruebas variadas bajo responsabilidad del propio fabricante que habrán eliminado la mayoría, si no la totalidad, de los errores.

SPICUM servicio de publicaciones

proceso de pruebas de interoperatividad nos la ofrece la organización EWOS[3]: la interoperatividad es "*la capacidad de dos o más sistemas abiertos para llevar a cabo una tarea común distribuida que es soportada por los protocolos OSI*" [ETG 029]. ETSI, conjuntamente, con EWOS definió una clasificación de la interoperatividad [ETR 130] [4].

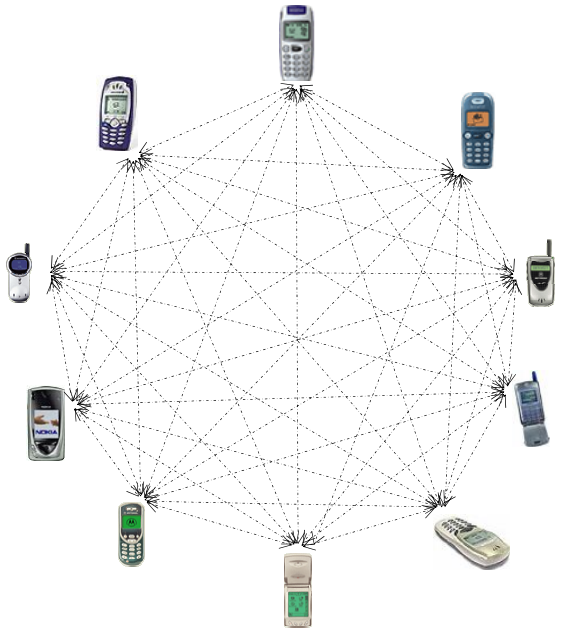


Figura 2.3: Esquema conceptual de las pruebas de interoperatividad.

Las pruebas de interoperatividad presentan varios problemas potenciales. Por un lado, el entorno de prueba puede no simular adecuadamente el entorno donde la IUT va a funcionar. Por otra parte, la interoperatividad no es transitiva, lo que origina que cada implementación deba enfrentarse a todas las demás para comprobar su funcionamiento (Figura 2.3). Si el número de suministradores es grande se puede convertir en un serio inconveniente para este enfoque. Por último, puede que el entorno de prueba no permita

[3] EWOS (*European Workshop for Open Systems*) fue sustituido en 1997 por ISSS (*Information Society Standardization System*). Hasta su desaparición fue el organismo responsable de potenciar la interoperabilidad de las implementaciones de sistemas basados en OSI comercializadas en Europa mediante la definición de los subconjuntos de opciones a utilizar para conseguir una funcionalidad determinada [IFLA92]. Entre otros documentos generados por este organismo se encuentran un recopilatorio de terminología aplicable a pruebas de conformidad y un análisis de distintas metodologías de pruebas de conformidad y su aplicación a perfiles [ETG 059]

[4] ETSI, conjuntamente con EWOS, definió una clasificación de la interoperatividad [ETR 130]. Según esta clasificación, existen cuatro categorías:

a) Interoperatividad de protocolos: capacidad de intercambiar mensajes a través de la plataforma de comunicaciones.

b) Interoperatividad de servicios: capacidad de soportar un subconjunto del servicio distribuido.

c) Interoperatividad de aplicación: capacidad de ofrecer una implementación consistente de la sintaxis y la semántica de los datos.

d) Interoperatividad percibida por el usuario: capacidad del usuario del servicio (humano o no) de intercambiar información a través del sistema.

SPICUM servicio de publicaciones

introducir errores en el sistema, impidiendo comprobar las respuestas frente a estímulos no válidos.

A cambio, la interoperatividad ofrece algunas ventajas frente a la conformidad. En primer lugar, puede ser más barata. Este coste dependerá, fundamentalmente, del número de implementaciones a probar y de la cobertura de las pruebas. Además, las pruebas de interoperatividad son más adecuadas para comprobar aquella funcionalidad que no aparezca en el estándar o que haya sido excluida de las pruebas de conformidad. Finalmente, los equipos para pruebas de conformidad pueden contener errores en su construcción.

Las pruebas de interoperatividad se están convirtiendo en un complemento de las pruebas de conformidad y su utilización está creciendo [EG 202 307]. Este tipo de pruebas se está empleando para demostrar el funcionamiento de productos en las etapas iniciales de introducción de los sistemas de comunicación, tiempo en el cual todavía no suele disponerse de Sistemas de Pruebas de conformidad. Además, se están usando también para comprobar funcionalidades no cubiertas por las pruebas de conformidad, como por ejemplo ocurre en el caso del sistema Bluetooth [BTEST PRO].

2.2 Descripción de la Metodología de Pruebas de Conformidad

El trabajo para la definición de una metodología de pruebas se inició en 1984 con el nombre de "Metodología y Marco de las Pruebas de Conformidad" (CTMF – *Conformance Testing Methodology and Framework*). En ese momento se disponía ya de abundante conocimiento y experiencia práctica ([SARI89], [LINN89], [LAPE88], [NAIK92], [ZENG89]) como resultado de varios años de investigaciones y el resultado de esta actividad fue el estándar ISO 9646 [ISO 9646]. Esta norma consta en la actualidad de 7 partes cuyo contenido se describe en la Tabla 2.1. Aunque su origen se encuentra dentro del estándar OSI ([BUSH90] describe un ejemplo de uso), su aplicación no está restringida al modelo de referencia OSI, sino que es aplicable a cualquier sistema o entidad de comunicaciones.

El objetivo de esta metodología es permitir la comparabilidad y aceptación de los resultados obtenidos por los laboratorios de pruebas, tanto por parte de los vendedores como por parte de sus clientes. Esto sólo se podía lograr por medio de métodos de certificación plenamente aceptados, idea que se recogió en dos medidas concretas: la elaboración de una metodología estandarizada de pruebas y el desarrollo de conjuntos de pruebas para los diferentes protocolos. Dichos conjuntos de pruebas deben ser elaborados en el marco de la metodología, que define tanto los métodos y procedimientos de prueba como la notación a emplear.

Se trata de ofrecer a los compradores una imagen consistente de calidad de los productos, garantizando en lo posible su correcta operación. Aunque la certificación supone un incremento de costes, ello redunda tanto en beneficio del cliente, que obtiene una mayor seguridad del producto que compra, como del propio suministrador, que ve respaldada la calidad de sus productos.

Existen numerosos aspectos que no han sido tenidos en cuenta dentro del estándar: las pruebas de interoperatividad, la acreditación de los laboratorios de pruebas, la medición de la robustez de una implementación, etc. Algunos de estos aspectos están siendo considerados en otros proyectos; otros, por el contrario, carecen actualmente de un marco de referencia.

SPICUM servicio de publicaciones

Cuando la metodología CTMF se combina con el uso de técnicas de descripción formal (FDT – *Formal Description Technique*), se denomina "Métodos Formales en Pruebas de Conformidad" (FMCT – *Formal Methods on Conformance Testing*) ([TRET94], [CAVA94], [WEZE91]). Mientras que CTMF extrae los casos de prueba de forma manual a partir de la especificación en lenguaje natural[5] y se basa en gran medida en la experiencia de los diseñadores de las pruebas, FMCT automatiza la generación de las secuencias de prueba, pudiendo, en teoría, generar secuencias óptimas. [SANG97] y [UYAR90] describen ejemplos de uso de la metodología FMCT sobre los protocolos INAP [Q.1214] y Q.931 [Q.931].

Tabla 2.1: Contenido de las partes de la norma ISO 9646.

Estándar[6]	Descripción
ISO 9646-1 X.290	Proporciona una visión general sobre el proceso de prueba y define conceptos relacionados con este campo.
ISO 9646-2 X.291	Enuncia una serie de requisitos acerca de la especificación de los Juegos de Pruebas Abstractas, ofreciendo unas guías sobre su elaboración y la elección del Método de Pruebas más apropiado.
ISO 9646-3 X.292	Define la notación de pruebas TTCN[7].
ISO 9646-4 X.293	Está relacionada con el proceso de realización de las pruebas.
ISO 9646-5 X.294	Describe esencialmente las relaciones entre el laboratorio de pruebas y el cliente. Lo hace en forma de documentos.
ISO 9646-6 X.295	Extiende la metodología de pruebas a los perfiles de protocolos, aunque las ideas se encuentran distribuidas en diversas partes de la norma.
ISO 9646-7 X.296	Especifica los requisitos respecto al diseño y uso de las proformas de conformidad.

2.2.1 Visión General

La Metodología de Pruebas de Conformidad de OSI está relacionada con las diferentes fases y actividades del proceso de prueba. Define no sólo el proceso para elaborar las pruebas, sino también los documentos que constituyen la entrada y salida de cada actividad, los Métodos de Pruebas y el proceso de evaluación realizado por un laboratorio de pruebas. Implícita se encuentra la idea de que, por razones de coste económico, las pruebas de conformidad deben ser limitadas.

Todo producto que desee manifestar que es conforme a una determinada especificación (estándar o no) debe ser certificado por un laboratorio de pruebas homologado. Para realizar esta certificación se definen un conjunto de pruebas que el producto debe superar y unos procedimientos que deben seguirse para realizarlas. El equipo que ejecuta las pruebas y genera un veredicto se denomina Sistema de Pruebas. La acreditación de un producto se obtiene cuando, enfrentado al Sistema de Pruebas, supera las pruebas que le son de aplicación.

[5] Con las imperfecciones asociadas a este tipo de lenguajes (ambigüedad, inconsistencia e incompletitud).

[6] ITU también ha publicado estas normas, cada parte en un documento, con la numeración X.290 a X.296.

[7] La descripción de esta notación no es objeto de este documento.

SPICUM servicio de publicaciones

La realización sometida a las pruebas se denomina Implementación Bajo Prueba (IUT – *Implementation Under Test*). La Implementación Bajo Prueba puede no encontrarse aislada, sino que puede formar parte de un sistema al cual se denomina Sistema Bajo Prueba (SUT – *System Under Test*). La funcionalidad de una Implementación Bajo Prueba se encuentra descrita en las Especificaciones de Base y en sus perfiles. Según ETSI [ETS 300 406], una Especificación de Base es "*la especificación de un protocolo, un servicio de telecomunicación, una interfaz, una sintaxis abstracta, unas reglas de codificación o unos objetos de información*" mientras que un perfil es la definición de un "*conjunto consistente de opciones de las Especificaciones de Base para proporcionar una cierta funcionalidad en un entorno dado*" [ETS 300 406]. Al conjunto de Especificaciones de Base y perfiles lo denominaremos Especificaciones de Sistema.

Las pruebas pueden definirse, y ejecutarse, para las Especificaciones de Base del sistema o para un perfil del mismo. En el segundo caso, el desarrollo de las pruebas se basa en las especificaciones previamente existentes para las pruebas de las Especificaciones de Base.

La primera actividad de la metodología de pruebas de conformidad parte del punto en el que las Especificaciones de Sistema se encuentran ya fijadas (ver Figura 2.4). Consiste en la elaboración de un conjunto de Propósitos de Prueba, es decir, requisitos que la Implementación Bajo Prueba debe satisfacer. Posteriormente, se define el Método de Pruebas (ATM – *Abstract Test Method*) o configuración estática del Sistema de Pruebas que se va a emplear para realizar cada una de las pruebas.

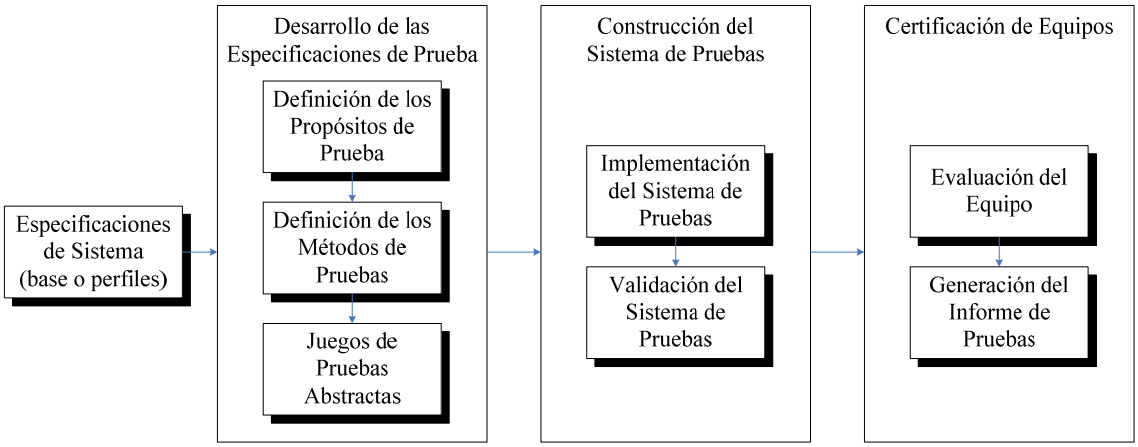


Figura 2.4: Proceso de la Metodología de Pruebas de Conformidad.

A continuación se define la secuencia de estímulos con que excitar a la Implementación Bajo Prueba, así como las respuestas esperadas a los mismos. En el caso de las pruebas de protocolo, la especificación de estas secuencias de eventos se hace formalmente mediante la notación TTCN ([X.292], [ES 201 873-1]).

A partir de las Especificaciones de Prueba, los suministradores de Sistemas de Pruebas, que en la metodología constituyen una entidad independiente de los suministradores de equipos del sistema de comunicaciones considerado [FUJI94], desarrollan los Sistemas de Pruebas correspondientes. Para lograr que estos Sistemas de Pruebas realmente realicen la función que se espera de ellos suele ser necesario que sus suministradores participen de forma activa en la definición de las Especificaciones de Prueba.

SPICUM servicio de publicaciones

Finalmente, cuando un suministrador desee poner a la venta un equipo, solicitará de un laboratorio de pruebas homologado la correspondiente certificación. Para ello, rellenará los documentos que definen la funcionalidad del equipo y recibirá un Informe de Pruebas. El formato y contenido de todos estos documentos es también parte de la metodología.

¿Cómo se comprueba que un Sistema de Pruebas realiza correctamente las mismas? Este problema, que puede llevar a una recursión infinita, se solventa aceptando que el proceso de validación a que son sometidos los Sistemas de Pruebas eliminan un elevado porcentaje de posibles ambigüedades, inconsistencias y errores. Durante la certificación de los equipos podrán surgir otros errores o incertidumbres, pero al realizarse las pruebas en un entorno controlado como son los laboratorios de pruebas, se resolverán fácilmente mediante consultas con los especificadores del sistema. Con el tiempo, preferiblemente en un corto plazo, surgirá un consenso en cuanto a la correcta operación de los equipos.

2.2.2 Juego de Pruebas Abstractas

Un Juego de Pruebas Abstractas (ATS – *Abstract Test Suite*) es una agrupación de pruebas, cada una de las cuales se denomina Caso de Prueba Abstracta (ATC – *Abstract Test Case*). La asociación se realiza, normalmente, en función del protocolo al cual es aplicable cada prueba[8]. Los requisitos necesarios para la estandarización de los Juegos de Pruebas Abstractas se encuentran recogidos en la parte 2 del estándar, donde también se incluyen unas ideas guía para su elaboración. Por simplicidad, a lo largo de este documento se utilizan los términos Juego de Pruebas Abstractas o Juego de Pruebas indistintamente; igualmente ocurre con el término Caso de Prueba Abstracta, al que se denota también Caso de Prueba o, simplemente, Prueba.

Un Caso de Prueba puede utilizar varios Componentes de Prueba, es decir, varias entidades que se ejecuten concurrentemente. Hay dos tipos: Componente de Prueba Principal (MTC – *Main Test Component*), que es único y se crea al inicio de la ejecución del Caso de Prueba, y Componente de Prueba Paralelo (PTC – *Parallel Test Component*), cero o más componentes creados por el Componente de Prueba Principal. La comunicación entre Componentes de Prueba se realiza a través de los Puntos de Coordinación (CP – *Coordination Point*).

A la hora de ejecutar un ATS, éste debe ser convertido en lo que se denomina un Juego de Pruebas Ejecutables (ETS – *Executable Test Suite*). El ETS es simplemente una versión ejecutable del modelo abstracto de las pruebas (el ATS), convenientemente adaptado a la plataforma de ejecución.

2.2.2.1 Estructura

La estructura de un Juego de Pruebas Abstractas es jerárquica (Figura 2.5-a), siendo la unidad independiente el Caso de Prueba Abstracta (Figura 2.5-b). Cada Caso de Prueba Abstracta tiene una finalidad determinada. Se reúnen en Grupos de Pruebas, en general, en función de algún criterio que defina el grupo, por ejemplo, las pruebas relacionadas

[8] Aunque una IUT puede contener uno o más protocolos de la misma capa o de varias capas adyacentes, el enfoque adoptado por OSI es llevar a cabo las pruebas de forma incremental, es decir, el protocolo más interno se prueba primero, y a continuación los siguientes en orden inverso de profundidad. La idea de realizar simultáneamente las pruebas sobre más de un protocolo fue abandonada para sistemas finales debido a las dificultades para escribir los correspondientes Juegos de Pruebas.

SPICUM servicio de publicaciones

con traspasos en un sistema UMTS. La modularidad se permite en forma de Pasos de Prueba (TS – *Test Step*), que son equivalentes al concepto de subrutina presente en numerosos lenguajes de programación. Tanto la estructura del Juego de Pruebas Abstractas como los propósitos particulares de cada Caso de Prueba se recogen en el correspondiente documento estandarizado denominado Estructura de los Juegos de Prueba y Propósitos de Prueba (TSS&TP – *Test Suite Structure & Test Purposes*) (ver sección 2.2.5.1).

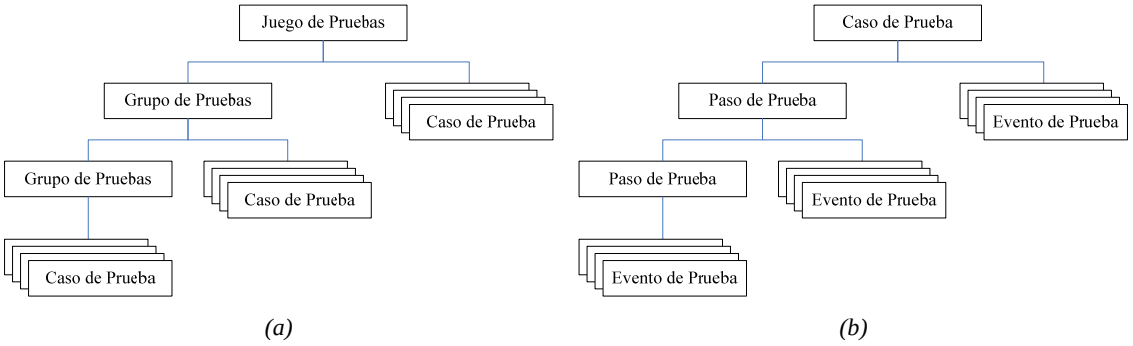


Figura 2.5: Estructura de un (a) Juego de Pruebas Abstractas y un (b) Caso de Prueba Abstracta.

El orden de ejecución de los Casos de Pruebas Abstractas es irrelevante para el proceso de certificación. Por ello es necesario que cada una de las pruebas termine en lo que se denomina un estado estable, es decir, un estado que no influya en la ejecución de los restantes Casos de Prueba. Así, conceptualmente, un Caso de Prueba Abstracta estará formado por:

- Prólogo: donde se sitúa a la IUT en el estado inicial de la prueba a partir del estado estable,
- Cuerpo: que lleva a cabo el propósito de la prueba y
- Epílogo: que permite alcanzar de nuevo el estado estable.

2.2.2.2 Veredictos

Para cada Caso de Prueba Abstracta contenido en un Juego de Pruebas Abstractas se obtiene un resultado al finalizar la ejecución del mismo. Hay tres veredictos posibles:

- ✓ Éxito (*Pass*): Indica que el resultado observado de la prueba satisface los requisitos de conformidad correspondientes al (o a los) propósito(s) del Caso de Prueba Abstracta.
- × Fracaso (*Fail*): En este caso algún requisito de conformidad no ha sido cumplido.
- ~ No concluyente (*Inconclusive*)[9]: Se obtiene este veredicto cuando no es posible asignar uno de éxito o fracaso porque, aunque el propósito de la prueba no se ha cumplido, no se ha encontrado ningún error. Por ejemplo, si ocurre una

[9] [CHAN93] plantea algunas situaciones en las que este veredicto puede emitirse y concluye que esta posibilidad debe minimizarse cuanto sea posible, proponiendo mecanismos para ello.

SPICUM servicio de publicaciones

liberación no esperada iniciada por los niveles inferiores. En [CHAN93] se concluye que este tipo de veredictos debe ser reducido al mínimo posible.

2.2.2.3 Tipos de Pruebas

Cuando se realiza una prueba el objetivo último es la comprobación de que la Implementación Bajo Prueba ofrece un comportamiento definido a priori mediante una especificación. Este comportamiento puede ser la ejecución de una o más acciones, un cambio de estado o, incluso, ninguna acción. Todo depende de si se busca verificar un comportamiento válido, la existencia de cierta funcionalidad o evidenciar la falta de respuesta ante estímulos erróneos. En la metodología definida por OSI las pruebas de conformidad se encuentran clasificadas en cuatro grupos dependiendo del objetivo de las mismas y su nivel de detalle. Estos grupos son los siguientes [X.290]:

a) Pruebas de interconexión básica: Se emplean para determinar si la IUT es capaz de establecer una interconexión o no. Son pruebas que deben realizarse al inicio del proceso. En caso de que su resultado sea negativo no es razonable continuar con el proceso de certificación.

b) Pruebas de capacidades: Verifican que las capacidades observables de la IUT cumplen los requisitos de conformidad estática y las capacidades indicadas en la Declaración de Conformidad de la Implementación (ver Sección 2.2.5.3) entregada por el cliente al laboratorio de pruebas.

c) Pruebas de comportamiento: Constituyen la parte principal de un Juego de Pruebas Abstractas. Comprueban el comportamiento de la IUT tan detalladamente como sea posible. Uno de sus objetivos es cubrir completamente el espectro de requisitos dinámicos de conformidad incluidos en la especificación. Se incluye aquí el análisis del comportamiento de la IUT frente a estímulos no válidos.

d) Pruebas de resolución de conformidad: Pruebas no normalizadas que permitan decidir la conformidad o no de la IUT con respecto a uno o más requisitos para los que no se ha definido un Caso de Prueba Abstracta. Son pruebas inadecuadas para determinar si una implementación posee o no una conformidad global.

2.2.2.4 Notación de Prueba

La notación de prueba ([X.292], [ES 201 873-1]) es independiente de los Métodos de Pruebas y su sintaxis ofrece distintas representaciones, siendo la representación tabular la más característica de ellas. La potencia de la notación está en que las construcciones básicas de la notación son el envío y recepción de eventos, incorporando per se mecanismos para observar el estado de las entradas y comparar los datos recibidos con un conjunto de alternativas. Proporciona también medios para definir primitivas y unidades de datos, así como una estructura para organizar los Casos de Pruebas incluidos en un Juego de Pruebas.

El comportamiento que no se puede modelar dentro de la notación de prueba, o para el cual no se dispone de mecanismos eficientes, se puede incluir mediante las Operaciones del Juego de Pruebas (TSOs – *Test Suite Operations*). Estas funciones se utilizan en general para programar algoritmos de propósito general, por ejemplo, calcular el CRC (*Cyclic Redundancy Check*) o cifrar una trama.

SPICUM servicio de publicaciones

2.2.3 Métodos de Pruebas Abstractas

Un Método de Pruebas Abstractas (ATM – *Abstract Test Method*) describe cómo probar la Implementación Bajo Prueba a un nivel de abstracción independiente de cualquier medio de prueba, pero con suficiente detalle como para permitir la especificación de Casos de Pruebas Abstractas [WETS06]. En concreto, un Método de Pruebas Abstractas especifica la configuración de una prueba, define las interfaces a través de las cuales el Sistema de Pruebas puede observar e influir el comportamiento de la IUT y proporciona algún mecanismo de coordinación de las pruebas.

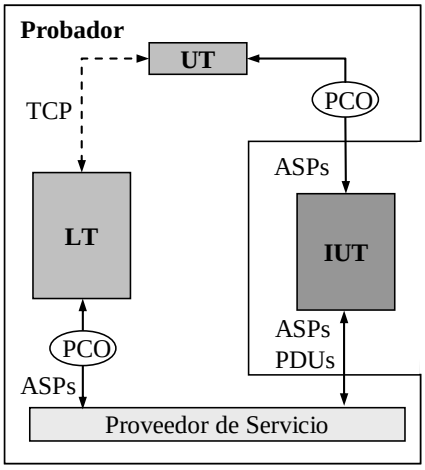


Figura 2.6: Esquema conceptual de la arquitectura de pruebas.

En un Método de Pruebas Abstractas existen los siguientes elementos (Figura 2.6):

- Probador Inferior (LT – *Lower Tester*): Controla y observa la frontera inferior de la IUT mediante el Proveedor de Servicio subyacente. Es el encargado de asignar el veredicto final de la prueba.
- Probador Superior (UT – *Upper Tester*): Controla y observa la frontera superior de la IUT. Para ello puede emplear un operador, una interfaz de programa o una interfaz hardware.
- Puntos de Control y Observación (PCO – *Point of Control and Observation*): Interfaces a través de las cuales el Sistema de Pruebas puede observar e influir el comportamiento de la IUT[10].
- Procedimientos de Coordinación de las Pruebas (TCP – *Test Control Procedures*): Reglas para la cooperación entre los Probadores Superior e Inferior.
- Proveedor de Servicio: Es la representación abstracta de la totalidad de las entidades que prestan un servicio [X.210], en este caso, entre el Probador Inferior y la IUT.
- Implementación Bajo Prueba (IUT): la realización que se desea probar.

[10] Un PCO se puede entender como dos colas de eventos, una en cada sentido. Sin embargo, la implementación concreta depende del fabricante.

SPICUM servicio de publicaciones

- Primitivas de Servicio Abstractas (ASP – *Abstract Service Primitive*) y Unidades de Datos del Protocolo (PDU – *Protocol Data Unit*): Primitivas de comunicación con la IUT y/o el Proveedor de Servicio y mensajes transportados por ellas, respectivamente.

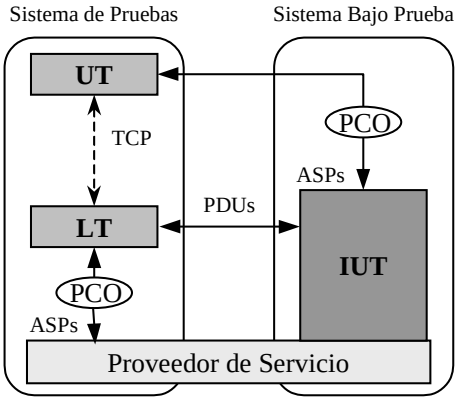


Figura 2.7: Arquitectura de pruebas para el método local.

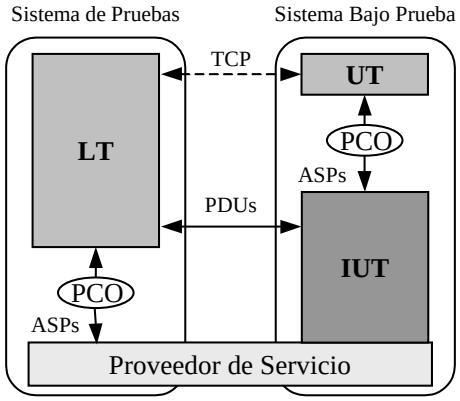


Figura 2.8: Arquitectura de pruebas para el método distribuido.

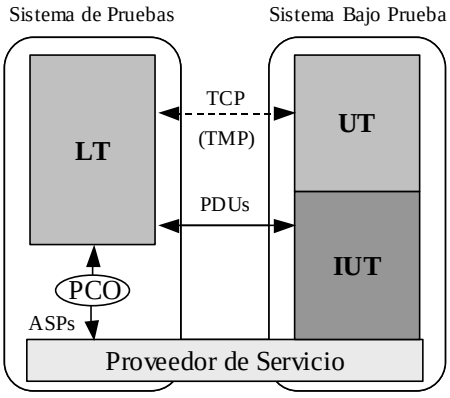


Figura 2.9: Arquitectura de pruebas para el método coordinado.

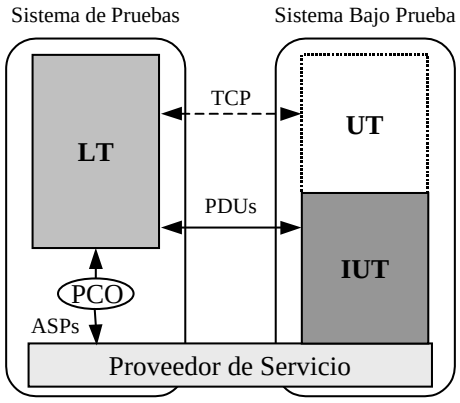


Figura 2.10: Arquitectura de pruebas para el método remoto.

A partir de una configuración genérica compuesta por los actores mencionados, la metodología deriva cuatro Métodos de Pruebas Abstractas específicos. Las diferencias entre ellos aparecen en su estructura, la ubicación del Probador Superior y, fundamentalmente, la capacidad de observación del comportamiento de la IUT durante la prueba. Estas configuraciones son las siguientes:

a) **Local** (Figura 2.7): El Probador Superior se encuentra en el Sistema de Pruebas. Emplea dos PCOs, uno bajo el Probador Inferior y otro bajo el Probador Superior. Requiere que la frontera superior de la IUT ofrezca una interfaz normalizada.

b) **Distribuido** (Figura 2.8): En este caso, el Probador Superior se encuentra dentro del Sistema Bajo Prueba. Utiliza un PCO bajo el Probador Inferior y otro bajo el Probador Superior. Este Método de Pruebas requiere que la frontera superior de

SPICUM servicio de publicaciones

la IUT sea una interfaz de usuario humano o una interfaz de lenguaje de programación normalizada.

c) **Coordinado** (Figura 2.9): Hace uso de tan sólo un PCO, situado debajo del Probador Inferior. El Probador Superior se encuentra situado dentro del Sistema Bajo Prueba y la coordinación se realiza mediante un Protocolo de Gestión de Pruebas (TMP – *Test Management Protocol*) normalizado.

d) **Remoto** (Figura 2.10): Usa también un único PCO, situado debajo del Probador Inferior. No existe el Probador Superior, sino que el Sistema Bajo Prueba proporciona algunas de las funciones del Probador Superior.

Todos los Métodos de Pruebas Abstractas admiten dos variantes. Se denominan no empotrados si es posible acceder a la frontera superior de servicio de la IUT. Por el contrario, se denominan empotrados si el control y la observación son sólo posibles en la frontera superior de otros protocolos que se encuentren por encima de la IUT .

Además, la metodología establece una distinción entre las pruebas de parte única, en las que la IUT se comunica con un único equipo, y las pruebas de partes múltiples, donde la IUT se comunica concurrentemente con varios equipos. Esta Tesis se restringe al ámbito de las pruebas de parte única.

2.2.4 Proceso de Evaluación de Conformidad

La certificación de una IUT involucra más actividades que la mera ejecución de una secuencia de pruebas. Se denomina Proceso de Evaluación de Conformidad al "*proceso completo de realización de todas las actividades de pruebas de conformidad necesarias para poder evaluar la conformidad de una realización o de un sistema con una o más especificaciones OSI*" [X.290]. El responsable a lo largo de todo el proceso (Figura 2.11) es el laboratorio de pruebas. Este proceso se encuentra integrado por 3 fases:

a) Preparación: El cliente debe garantizar que la IUT puede ser probada mediante al menos un Método de Pruebas Abstractas. Para ello, laboratorio y cliente negociarán un acuerdo sobre cuál es la IUT y qué Método y Juegos de Pruebas se emplearán.

b) Operaciones de prueba: Incluyen la revisión de los requisitos estáticos de conformidad, la selección de las pruebas a ejecutar sobre la IUT, la parametrización de dichas pruebas y la realización de las pruebas obteniendo un veredicto que se comunicará al cliente a través del correspondiente Informe de Pruebas.

c) Generación de informes de pruebas: Es la fase final y consiste en la redacción de un Informe de Pruebas de Conformidad del Sistema y tantos Informes de Pruebas de Conformidad de Protocolo como protocolos hayan sido probados.

SPICUM servicio de publicaciones

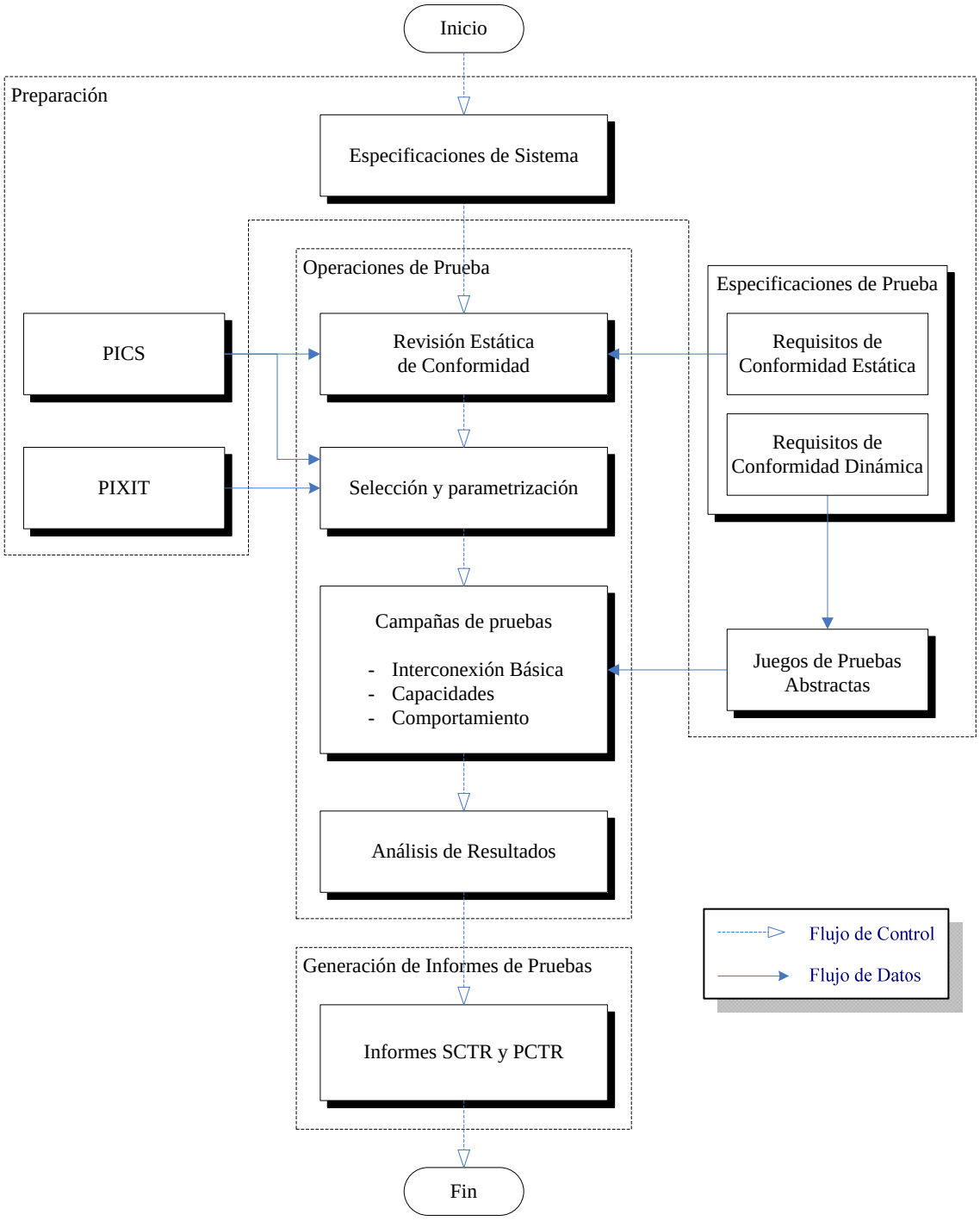


Figura 2.11: Visión general del proceso de certificación de la conformidad [ETG 059].

2.2.5 Documentos Utilizados en la Metodología

La Metodología de Pruebas formulada por OSI define no sólo los procedimientos para llevar a cabo la certificación de equipos, sino también un conjunto de documentos (Figura 2.12) que deben generarse (y rellenarse) en diferentes fases del proceso. A continuación se recoge una breve explicación de cada uno de estos documentos.

SPICUM servicio de publicaciones

2.2.5.1 Especificaciones de Prueba de Base

En primer lugar aparecen los documentos que definen las pruebas de conformidad para aquellos equipos que implementen la funcionalidad definida en las Especificaciones de Base [X.291]. Estos documentos son los fundamentales pues definen el alcance de las pruebas y determinan el grado de cobertura y fiabilidad que es posible esperar tras someter una IUT a su análisis.

- Estructura de los Juegos de Pruebas y Propósitos de Prueba (TSS&TP – *Test Suite Structure & Test Purposes*): Este documento contiene la estructura de los Juegos de Pruebas Abstractas, indicando qué Grupos de Pruebas existen y cuáles son los Casos de Pruebas Abstractas en cada uno de ellos, así como una descripción del Propósito de la Prueba, la secuencia de eventos esperada y el veredicto a asignar a la ejecución de la prueba en cada caso. Estas descripciones se realizan lenguaje natural.
- Juegos de Pruebas Abstractas (ATS – *Abstract Test Suite*): Incluye una descripción del Método de Pruebas Abstractas así como la definición, en notación formal TTCN, de cada uno de los Casos de Pruebas Abstractas descritos en el documento TSS&TP.

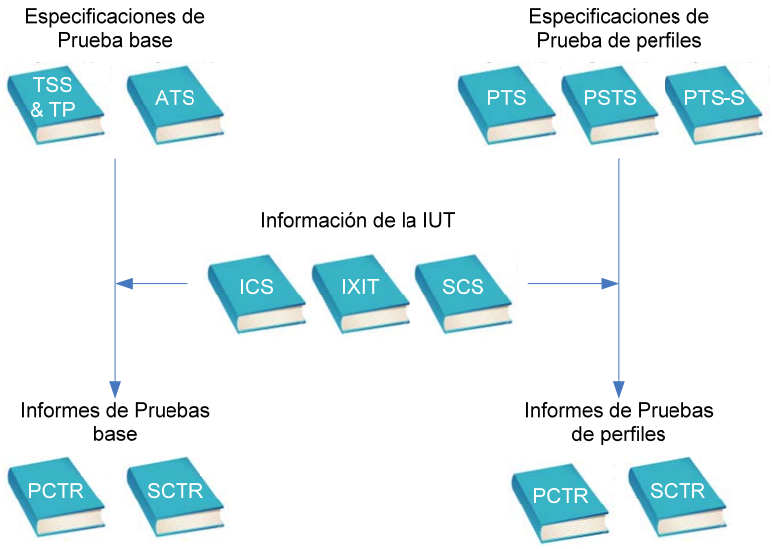


Figura 2.12: Relación entre los documentos definidos en la Metodología de Pruebas de Conformidad.

2.2.5.2 Especificaciones de Prueba de Perfiles

El desarrollo de las pruebas para perfiles se basa en las especificaciones existentes para las pruebas de las Especificaciones de Base. Los documentos relacionados con las pruebas de perfil [X.295] son:

- Especificación de Pruebas de Perfil (PTS – *Profile Test Specification*): Es el conjunto de todas las especificaciones de pruebas de conformidad necesarias para evaluar la conformidad de un perfil. Este documento incluye la Estructura y Propósitos de las Pruebas, los Juegos de Pruebas Abstractas relacionados y la Especificación de las Pruebas Específicas del Perfil. Es el documento equivalente al TSS&TP para el perfil.

SPICUM servicio de publicaciones

- Especificación de las Pruebas Específicas del Perfil (PSTS – *Profile Specific Test Specification*): Contiene los Casos de Pruebas Abstractas específicos del perfil. Estos Casos de Prueba pueden ser añadidos o modificados partiendo de los Casos de Prueba de las Especificaciones de Base. Es el documento equivalente al Juego de Pruebas Abstractas para el perfil.

2.2.5.3 Información de la Implementación Bajo Prueba

Para poder ejecutar las pruebas, el laboratorio de pruebas requiere información adicional sobre la IUT que debe proporcionar el suministrador. Esta información no es solamente una indicación de cuál es la funcionalidad del equipo, sino también parámetros necesarios para llevar a cabo las pruebas como, por ejemplo, un código de identificación del mismo. Los documentos pertinentes ([X.294], [X.296]) son los siguientes:

- Declaración de Conformidad de la Implementación (ICS – *Implementation Conformance Statement*): Contiene información acerca de la funcionalidad realizada por la IUT. Este documento será usado tanto en la revisión de los requisitos estáticos de conformidad como en la selección y parametrización de los Casos de Prueba. Si la revisión estática diera un resultado negativo, la IUT no llegaría a la etapa de verificación frente al Sistema de Pruebas. En el caso de protocolos el formulario se denomina PICS (*Protocol ICS*).
- Información Suplementaria de la Implementación para las Pruebas (IXIT – *Implementation eXtra Information for Testing*): Contiene información que complementa la recogida en la ICS. Está relacionada con los Juegos de Pruebas Abstractas, las Especificaciones de Pruebas de Perfiles y los Métodos de Pruebas Abstractas. En el caso de protocolos se denomina PIXIT (*Protocol IXIT*). Es información necesaria para el correcto funcionamiento, pero no relacionada con su funcionalidad como, por ejemplo, la clave de autentificación para acceder a un determinado servicio.
- Declaración de Conformidad del Sistema (SCS – *System Conformance Statement*): Recoge un índice de las especificaciones con las que la IUT alega ser conforme, de las Declaraciones de Conformidad de la Implementación y de los Informes de Prueba relevantes.

2.2.5.4 Informes de Pruebas

Como resultado del proceso de verificación, el laboratorio de pruebas es responsable de entregar al suministrador del equipo los siguientes Informes de Pruebas [X.294]:

- Informe de Pruebas de Conformidad del Sistema (SCTR – *System Conformance Test Report*): Incluye información administrativa y un resumen de los resultados de las pruebas, dejando claro si el equipo ha pasado las pruebas o no. En caso de certificar varios perfiles, debe entregarse un SCTR por cada perfil probado.
- Informe de Pruebas de Conformidad del Protocolo (PCTR – *Protocol Conformance Test Report*): Contiene, para un protocolo concreto, el resultado de todos los Casos de Prueba ejecutados así como cuantos registros de conformidad (trazas de ejecución de las pruebas) solicite el cliente. Debe entregarse un PCTR por cada protocolo probado.

SPICUM servicio de publicaciones

2.2.6 Desarrollo de un Juego de Pruebas Abstractas

La metodología OSI propone las siguientes actividades, sin un orden predefinido, para desarrollar un Juego de Pruebas Abstractas:

a) Determinar los requisitos de conformidad que pueden ser comprobados.

b) Determinar los Grupos de Pruebas y sus objetivos para cubrir los anteriores requisitos.

c) Desarrollar los Propósitos de las Pruebas.

d) Elegir un Método Abstracto de Pruebas.

e) Especificar los Casos de Pruebas Abstractas.

f) Especificar los criterios de selección respecto a los documentos relacionados con la implementación que debe entregar el suministrador (ICS e IXIT).

El diseño de un Juego de Pruebas Abstractas requiere un conocimiento de la especificación tan profundo como si se fuera a llevar a cabo su implementación. De otra forma, el resultado puede convertirse en papel mojado, incapaz de cumplir su función de certificación. El balance entre cobertura y coste económico de las Pruebas debe ser tenido en cuenta en todo momento ya que en numerosas ocasiones se puede estar tentado de incluir más pruebas de las necesarias para obtener una confianza razonable del correcto funcionamiento de la implementación, tal vez para lograr un escaso incremento de dicha confianza.

2.3 Conclusión

La Metodología de Pruebas de Conformidad proporciona unas líneas generales para avanzar en el proceso de evaluación del funcionamiento de una implementación con respecto a su especificación. Trata los aspectos de la generación de los Juegos de Pruebas Abstractas, el proceso de ejecución de las mismas, las relaciones entre el laboratorio de pruebas y el suministrador, los documentos necesarios a lo largo de todo el proceso, tanto de entrada como los informes de resultados que se generarán, y estandariza una notación para el desarrollo de los Juegos de Pruebas Abstractas.

Esta metodología ha servido como base tanto para la generación de Juegos de Pruebas Abstractas de numerosas especificaciones como para la aceptación de los resultados de las pruebas por todos los actores implicados en el proceso. Aunque fue desarrollada dentro del marco OSI, su ámbito es mucho más amplio, pudiendo aplicarse a cualquier implementación de un sistema de comunicaciones. Como primer paso dado en este campo, esta metodología no trata todos los aspectos relacionados con el proceso de pruebas. Fuera de ella se encuentran, por ejemplo, los temas de las pruebas de interoperatividad, las prestaciones de la IUT o el proceso de acreditación de los laboratorios de pruebas.

SPICUM servicio de publicaciones

SECCIÓN I

METODOLOGÍA

SPICUM
servicio de publicaciones

SPICUM
servicio de publicaciones

En esta Sección se describe una Metodología de Diseño para Sistemas de Pruebas basada en el uso de lenguajes ITU, una arquitectura genérica y un conjunto de herramientas de soporte, comerciales y propias.

En primer lugar se presenta la arquitectura. Ésta es independiente de la plataforma de ejecución y ha sido pensada para poder reutilizar un alto porcentaje de sus componentes entre distintos Sistemas de Pruebas. La separación en Subsistemas claramente diferenciados ofrece una alta flexibilidad, permitiendo distintas configuraciones en su despliegue. Esta arquitectura es válida para todo tipo de pruebas.

La Metodología de Diseño está compuesta de ocho fases, cada una de las cuales realiza una o más actividades que transforman los modelos a su entrada. La Metodología parte de las Especificaciones de Sistema y Especificaciones de Prueba y, a partir de ahí, engloba todas las actividades necesarias para el diseño de un Sistema de Pruebas. La aplicación de la Metodología es independiente de herramientas comerciales específicas, pero requiere de ellas para una utilización eficiente.

Las herramientas de soporte a la Metodología de Diseño son de dos tipos: componentes software genéricos, que son utilizados en unión con otros componentes, y generadores automáticos, que transforman información en componentes software. Estas herramientas complementan a los entornos comerciales de desarrollo. Parte de las herramientas construidas son estáticas, mientras que otras deben ser particularizadas para cada Sistema de Pruebas específico. En este último grupo se ha intentado proporcionar la mayor automatización posible, aunque en ocasiones esto no es posible. Para el diseño de estas herramientas se ha puesto el máximo énfasis en la portabilidad de sus producciones.

Las pruebas de conformidad de radio y de protocolos se han considerado, tradicionalmente, mundos distantes. Una diferencia esencial es que las primeras requieren instrumentación específica capaz de llevar a cabo cierto conjunto de medidas radioeléctricas sobre la interfaz aire. No obstante, la mayoría de los conceptos son compartidos. Por ello, se propone una metodología para el diseño de Sistemas de Pruebas radio que acerque este campo al nivel alcanzado por las pruebas de protocolo. Esta asimilación permite emplear todas las herramientas ya existentes en el área de las pruebas de protocolos y construir elementos comunes para ambos tipos de pruebas.

Por último, se describe la evaluación de la metodología SOMT, que utiliza como lenguaje preferente SDL, cuando es aplicada al diseño de sistemas basados en estándares. Como resultado de este estudio se han propuesto modificaciones de la metodología para su adaptación a procesos de diseño similares al analizado. La metodología resultante se ha denominado M-SOMT (*Modified SOMT*).

SPICUM servicio de publicaciones

SPICUM
servicio de publicaciones

Capítulo

3

"Otros hábitats se amoldaban a diseños menos pragmáticos. Había espirales salvajes y hélices, como cristal soplado o conchas de nautilus. Había enormes concatenaciones de esferas y tubos que se asemejaban a moléculas orgánicas. Había hábitats que se rediseñaban a sí mismos continuamente, lentas sinfonías de movimiento de arquitectura pura. Había otros que se habían aferrado a un diseño pasado de moda a lo largo de siglos de tozudez, resistiendo cualquier innovación y falsa apariencia. Otros pocos se habían camuflado en nebulosas de materia pulverizada, ocultando su verdadero diseño."

Xavier Liu, El Arca de la Redención

CAPÍTULO 3: ARQUITECTURA PARA SISTEMAS DE PRUEBAS

La Metodología de Pruebas de Conformidad define el proceso de certificación de un equipo y las entidades que participan en el mismo. Este proceso debe ser adaptado para cada sistema, con la selección de los Propósitos de las Pruebas y la elaboración de los Juegos de Prueba, en TTCN, que realizan dichos Propósitos. Sin embargo, bajo el paradigma de 'sistemas abiertos', la Metodología deja en libertad a los suministradores para construir los Sistemas de Pruebas como consideren más adecuado con la única restricción de que estos deben realizar fielmente la semántica de los Casos de Prueba, así como las interfaces con el resto de componentes.

El desarrollo de un Sistema de Pruebas es un problema de Ingeniería de Sistemas que engloba múltiples disciplinas: protocolos, software, hardware, procesado de señal, radio. En su realización intervienen distintos ingenieros con conocimientos muy variados. La forma de atacar esta complejidad es particionando el problema en tareas más pequeñas, con interfaces claramente definidas.

A alto nivel, un Sistema de Pruebas se puede ver como una plataforma de ejecución configurable, que se acondiciona según las características de los Casos de Prueba a realizar. Gran parte de la funcionalidad se puede entender como de soporte, en el sentido de que realiza tareas que no dependen del Sistema Bajo Prueba. Mediante un diseño adecuado, es posible agrupar esta funcionalidad en módulos reutilizables, separando las funciones comunes de las partes específicas.

SPICUM servicio de publicaciones

En este capítulo se presenta una arquitectura genérica independiente de la plataforma de ejecución y pensada para poder reutilizar un alto porcentaje de sus componentes entre distintos Sistemas de Pruebas. La separación en Subsistemas claramente diferenciados ofrece una alta flexibilidad, permitiendo distintas configuraciones en su despliegue. La división en módulos permite la modificación y mejora de uno o más componentes sin afectar al diseño de los demás. La funcionalidad de adaptación se ha segregado en módulos independientes y se ha incluido funcionalidad de traza para facilitar la depuración del sistema y su operación. Esta arquitectura es válida para todo tipo de pruebas.

Este capítulo se ha estructurado en dos partes. En la primera se ofrece una visión global de la arquitectura. En apartados posteriores, se describe cada uno de los Subsistemas que la componen.

3.1 Descripción de la Arquitectura

La arquitectura ha sido concebida con objeto de ofrecer una flexibilidad adecuada tanto al diseñador del Sistema de Pruebas como al operador del mismo, así como para satisfacer las restricciones impuestas por los distintos Métodos de Pruebas (ver Capítulo 2, Sección 2.2.3) posibles en la Metodología de Pruebas de Conformidad.

También hay algunos otros puntos que se deben tener en cuenta. En primer lugar, hay que considerar que la certificación de un equipo, por ejemplo, un teléfono DECT conlleva la realización de pruebas pertenecientes a distintos Juegos de Pruebas; sin embargo, el Informe de Pruebas final que debe emitirse debe ser único. Por tanto, la arquitectura debería ser suficiente flexible como para permitir cambiar el Juego de Pruebas manteniendo la visión global del proceso de certificación.

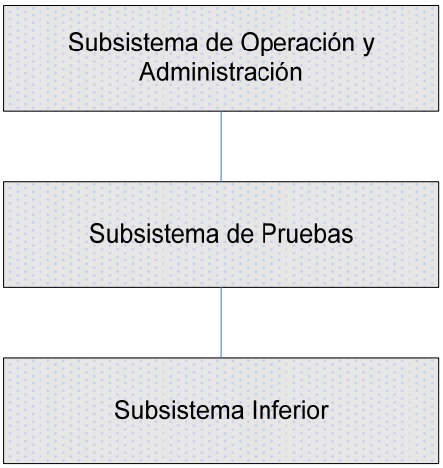


Figura 3.1: Subsistemas de que consta la Arquitectura de un Sistema de Pruebas.

En segundo lugar, el Sistema de Pruebas incluye sólo parte del Proveedor de Servicio, en concreto, los protocolos y componentes que sean necesarios para comunicar el Comprobador Inferior con el Sistema Bajo Prueba. A este subconjunto del Proveedor de Servicio se le va a denominar Subsistema Inferior. Las responsabilidades del diseño de los Juegos de Pruebas y del Subsistema Inferior corresponden a grupos distintos (organismo de estandarización, suministrador), y también utilizan lenguajes diferentes,

SPICUM servicio de publicaciones

ya que los Juegos de Pruebas se modelan en TTCN. Por ello, y con objeto de que modificaciones en uno no afecten al otro, estos elementos serán entidades independientes.

Y, por supuesto, el operador necesita de un mecanismo para controlar la ejecución de las pruebas, visualizar sus resultados y generar informes. A este elemento le denominaremos Subsistema de Operación y Administración.

Así, a un nivel muy abstracto, la arquitectura posee los Subsistemas indicados en la Figura 3.1. La funcionalidad de cada uno de estos Subsistemas es la siguiente:

- Subsistema de Operación y Administración: Permite al operador controlar la ejecución de las pruebas y gestionar sus resultados. Es un elemento externo al Método de Pruebas.
- Subsistema de Pruebas: Incluye las pruebas y todos los elementos necesarios para su ejecución. Representa lo que se denomina el Ejecutable de Pruebas en TTCN-3 [ES 201 873-5]. Contiene el Probador Inferior (LT) y el Probador Superior (UT).
- Subsistema Inferior: Contiene los protocolos y elementos que hacen falta para comunicar el Subsistema de Pruebas con el Sistema Bajo Prueba. Realiza una función equivalente a lo que en [ES 201 873-5] se denomina el Adaptador del Sistema Bajo Prueba (*SUT Adaptor*). Engloba la parte del Proveedor de Servicio que se encuentra dentro del Sistema de Pruebas.

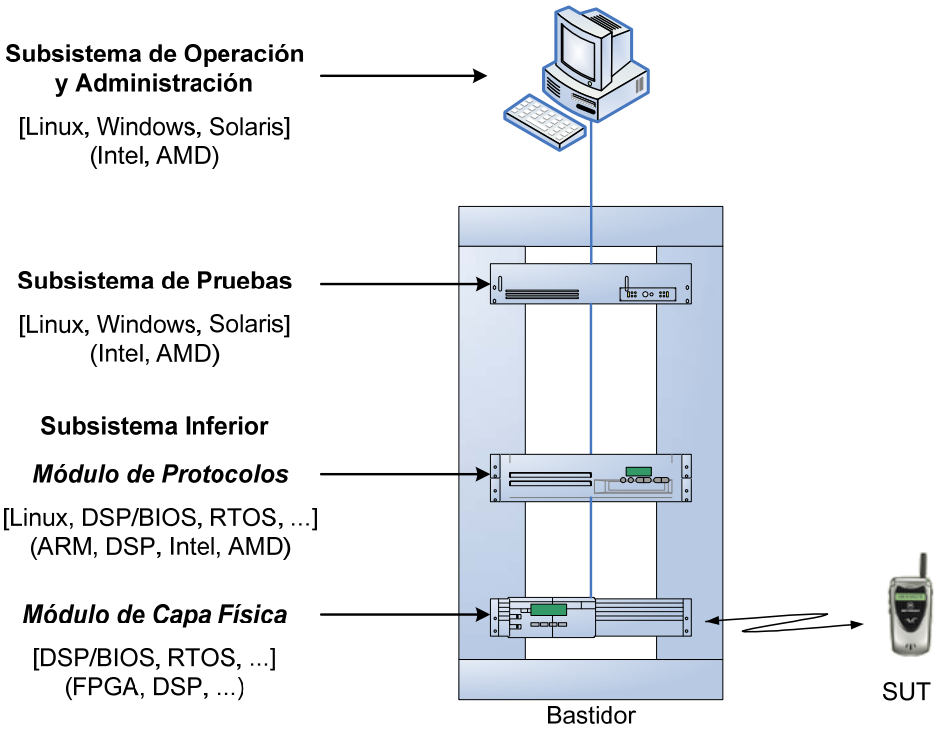


Figura 3.2: Distribución flexible de Subsistemas entre elementos físicos.

La interfaz que ofrece el Sistema de Pruebas al Sistema Bajo Prueba puede ser una interfaz software o una interfaz eléctrica. En el caso de los sistemas de comunicaciones móviles se trata de una interfaz radio, por lo que el Subsistema Inferior incluye desde

SPICUM servicio de publicaciones

protocolos de Nivel 3 hasta el procesado de señal que un transmisor o un receptor deben realizar en el nivel físico. Por ello, este Subsistema se ha estructurado en dos módulos: Módulo de Protocolos y Módulo de Capa Física. La separación de la funcionalidad entre uno y otro se basará en un criterio de sincronización temporal.

Esta estructura es genérica, y permite certificar cualquier implementación de cualquier tecnología, cargando los subsistemas apropiados. Cada Subsistema se modela como una entidad independiente (dos en el caso del Subsistema Inferior), pudiendo asignarse a plataformas distintas. Por ejemplo, dado que el Subsistema Inferior incluye el Nivel Físico, tendrá unas mayores restricciones temporales a la hora de procesar paquetes y generar sus respuestas. En este caso, una opción sería fabricar una plataforma con una tarjeta especialmente dedicada a ejecutar el Subsistema Inferior. Además, el Subsistema de Operación y Administración no tiene por qué encontrarse físicamente al lado del Subsistema de Pruebas[1], permitiendo que un operador controle remotamente la ejecución de los Casos de Prueba; por ejemplo, en el caso de una cámara anecoica o con temperatura controlada. Así, la arquitectura permite una asignación de Subsistemas a plataformas físicas como la mostrada en la Figura 3.2.

A lo largo de esta Tesis se va a utilizar un código de colores y formas para clasificar los componentes un Sistema de Pruebas. Para realizar esta clasificación se han tenido en cuenta tres criterios: dependencia de la Implementación Bajo Prueba, dependencia de la plataforma de ejecución y disponibilidad al inicio del diseño. Los dos primeros se representan en las figuras de la arquitectura mediante colores (izquierda de la Figura 3.3). Se ha diferenciado entre componentes dependientes e independientes de la Implementación Bajo Prueba y elementos dependientes e independientes de la plataforma de ejecución; los elementos dependientes se representan con un color más oscuro. El grado de disponibilidad se indica mediante su forma (derecha de la Figura 3.3). Se ha distinguido entre componentes disponibles previamente a este trabajo (estándares, herramientas comerciales), componentes que han sido implementados (herramientas propias) y componentes que son específicos de cada Sistema de Pruebas (modelos abstractos, codificadores).

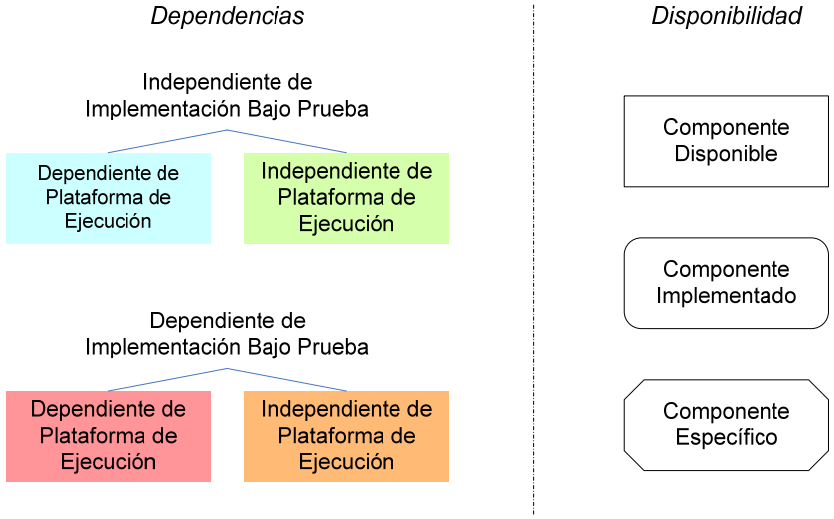


Figura 3.3: Leyenda de colores y formas empleados en los elementos de un Sistema de Pruebas.

[1] Podría encontrarse también distribuido entre varios elementos físicos.

SPICUM servicio de publicaciones

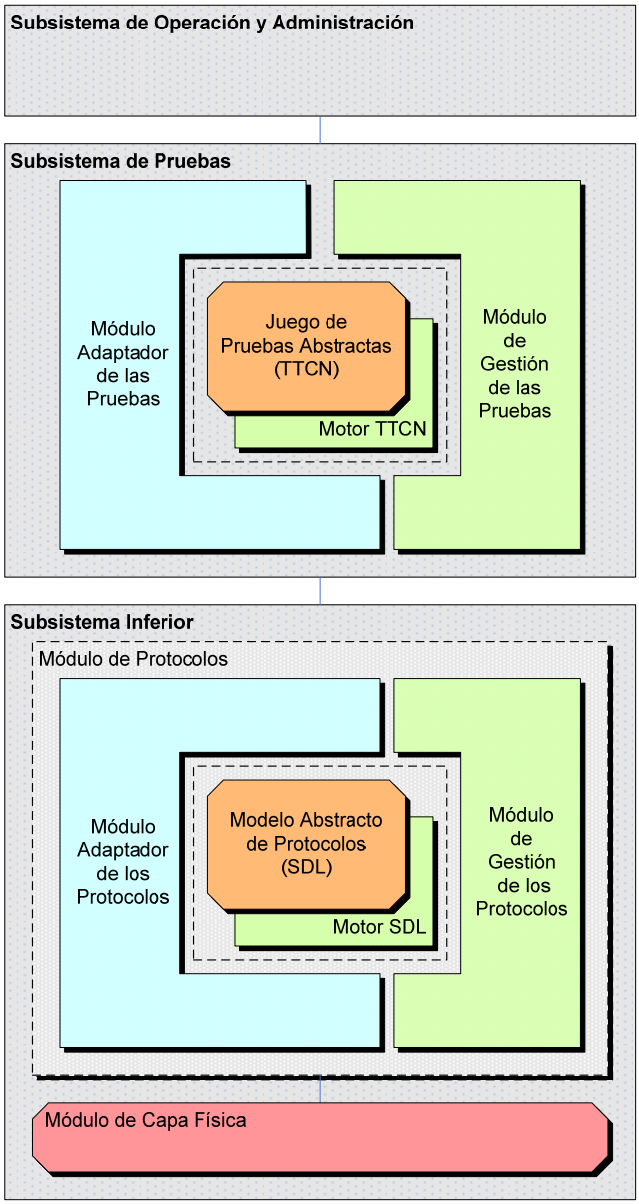


Figura 3.4: Esquema de Nivel Medio de la Arquitectura

En la Figura 3.4 se muestra un mayor nivel de detalle de la arquitectura de la Figura 3.1. Tanto el Subsistema de Pruebas como el Módulo de Protocolos se van a encontrar modelados en un lenguaje abstracto, es decir, independiente de la plataforma. Estos modelos son el Juego de Pruebas Abstractas (en TTCN) y el Modelo Abstracto de Protocolos (en SDL). Ambos necesitan un conjunto de librerías que interpreten la semántica del lenguaje, en tiempo de compilación, y planifiquen la secuencia de acciones a realizar. Por ello, la estructura de ambos subsistemas contiene un módulo que corresponde al modelo abstracto y otro módulo, que se ha denominado Motor, que corresponde a las librerías de interpretación de la semántica.

A la hora de su ejecución, es necesario adaptar los Modelos Abstractos y su Motor a la plataforma elegida. Los elementos necesarios se han agrupado en dos módulos, que se han denominado Módulo Adaptador, que envuelve al Juego de Pruebas Abstractas y le permite acceder a los servicios característicos de un sistema operativo (temporizadores,

SPICUM servicio de publicaciones

entrada/salida, …), y Módulo de Gestión, que proporciona servicios de alto nivel (codificación/descodificación de primitivas, control, …).

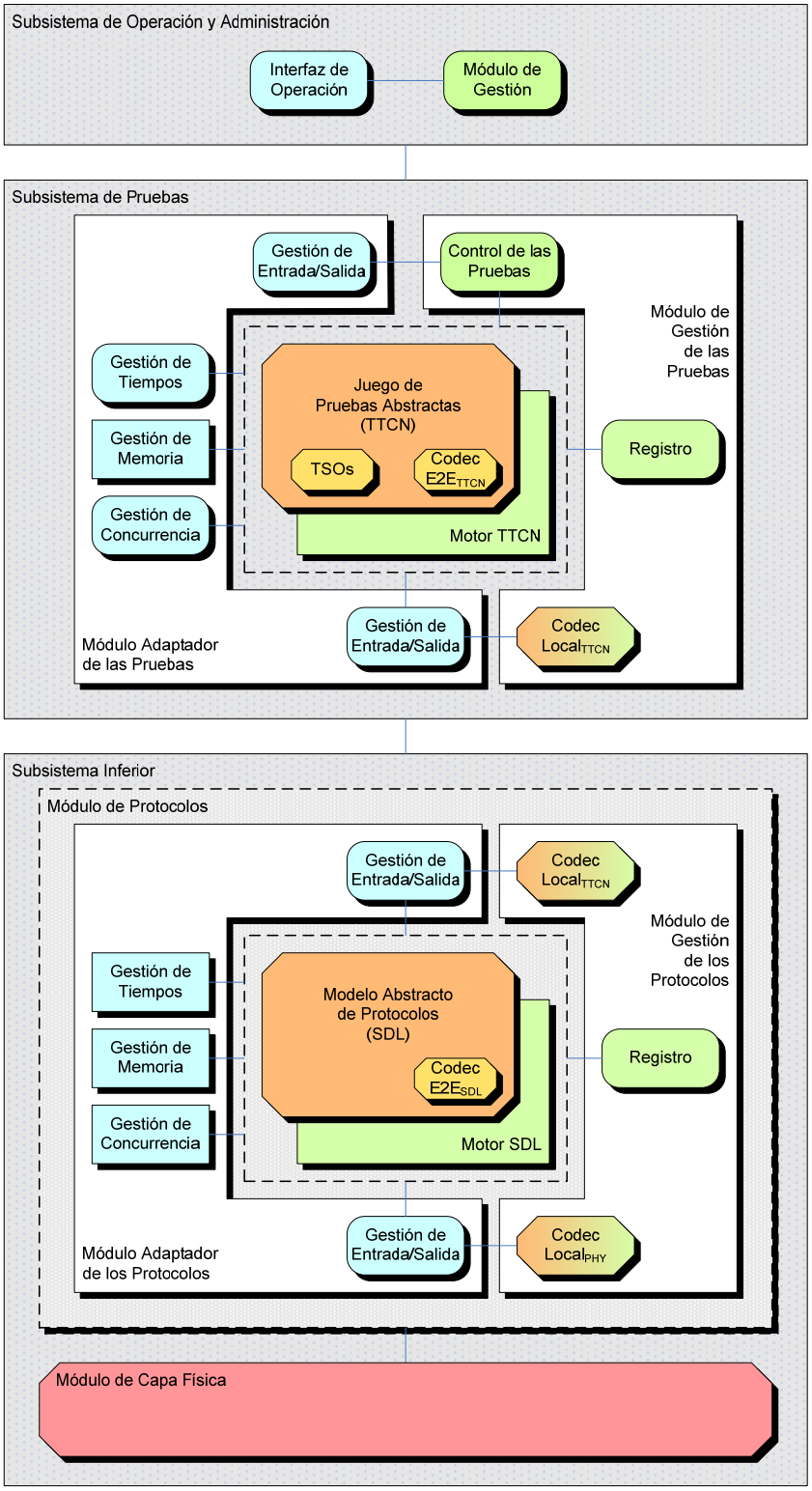


Figura 3.5: Arquitectura detallada de un Sistema de Pruebas.

SPICUM servicio de publicaciones

Finalmente, identificando los componentes de cada Módulo Adaptador y de Gestión, y estructurando el diseño del Subsistema de Operación y Administración, se llega a la estructura detallada (Figura 3.5). Esta estructura se explica, para cada Subsistema, en la siguiente Sección. Los componentes Codec $Local_{TTCN}$ y Codec $Local_{PHY}$ se han pintado con un color degradado porque su implementación utiliza tanto partes independientes de la IUT (genéricas) como partes dependientes de la IUT (que se construyen para cada Sistema de Pruebas).

Se han construido todos los elementos genéricos que no se encontraban disponibles (componentes propios), así como herramientas para la generación automática de los codificadores entre el Subsistema de Pruebas y el Subsistema Inferior (Codec $Local_{TTCN}$). Estas implementaciones se describen en el Capítulo 5. De esta forma, cuando se diseñe un nuevo Sistema de Pruebas sólo es necesario construir los elementos que son específicos para las nuevas pruebas. Estos elementos aparecen resaltados en la Figura 3.6.

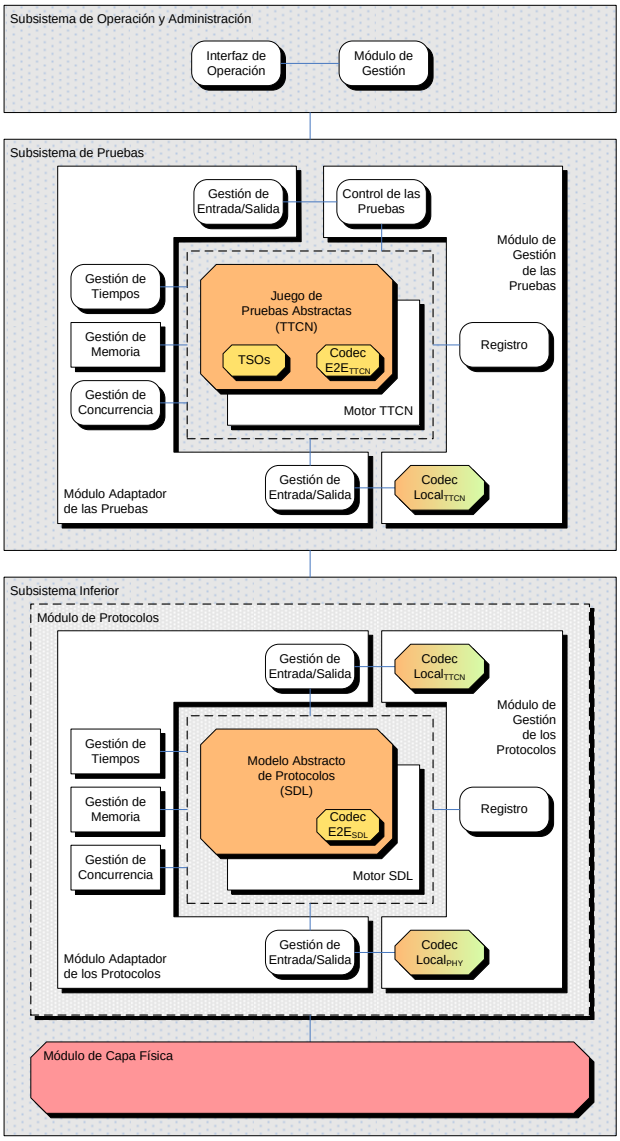


Figura 3.6: Elementos específicos de cada Sistema de Pruebas.

SPICUM servicio de publicaciones

3.2 Subsistemas de la Arquitectura

En los siguientes apartados se explica, en mayor detalle, la estructura de cada uno de los Subsistemas.

3.2.1 Subsistema de Operación y Administración

El Subsistema de Operación y Administración es el elemento que interacciona con el operador y debe permitir la ejecución y el control de las pruebas sobre una determinada Implementación Bajo Prueba, así como la visualización y gestión de los resultados. Se ha estructurado en dos componentes (Figura 3.7)[2]:

- Interfaz de Operación: Elemento gráfico a través del cual se comunica el operador del Sistema de Pruebas. Se ha considerado que tiene una alta dependencia con la plataforma donde se ejecute; sin embargo, existen opciones, como puede ser el lenguaje Java, para lograr una mayor independencia.
- Módulo de Gestión: Contiene la funcionalidad necesaria para la ejecución, control y gestión de las pruebas y sus resultados, así como para la generación de Informes de Pruebas de una IUT. Esta funcionalidad se ha considerado independiente de la plataforma de ejecución.

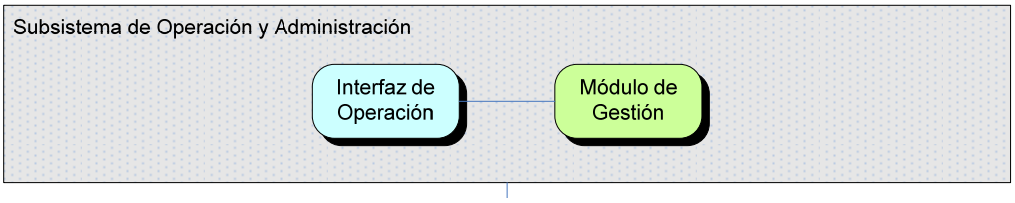


Figura 3.7: Estructura detallada del Subsistema de Operación y Administración.

La funcionalidad básica de este Subsistema es la siguiente:

- Selección del Juego de Pruebas Abstractas a utilizar.
- Edición y visualización de los Parámetros de Pruebas de la IUT.
- Identificación de los Casos y Grupos de Pruebas aplicables a una determinada IUT.
- Selección de uno o más Casos de Prueba.
- Ejecución de los Casos de Prueba seleccionados.
- Registro tanto de los veredictos de la ejecución como de las trazas generadas durante la misma. Estas trazas pueden resultar de gran ayuda a la hora de identificar posibles problemas tras la ejecución de las pruebas si los resultados no son los esperados.
- Posibilidad de abortar la ejecución del Caso de Prueba, o conjunto, cuya ejecución se encuentre en curso.

[2] Se puede considerar que un tercer componente de este Subsistema sea un módulo de adaptación a la plataforma (funciones de bajo nivel y de acceso al sistema operativo). Sin embargo, al no ser este Subsistema el objeto central de este trabajo y dado que no depende de la tecnología bajo prueba y es una herramienta implementada en su totalidad, esta adaptación no se muestra como un módulo independiente.

SPICUM servicio de publicaciones

- Visualización de las trazas generadas durante las ejecuciones (una o más) de cada prueba. Esta visualización debe ser posible tanto en modo texto como en modo gráfico mediante la representación del correspondiente diagrama de secuencia de mensajes.
- Filtrado de las trazas en función de su contenido.
- Posibilidad de volcar las trazas obtenidas a un fichero con formato MSC [Z.120].
- Generación de informes de certificación.

3.2.2 Subsistema de Pruebas

El Subsistema de Pruebas es el encargado de ejecutar las pruebas y emitir el veredicto. Corresponde a lo que se denomina el Ejecutable de Pruebas en [ES 201 873-5] y engloba al Probador Inferior (LT) y al Probador Superior (UT) (Figura 3.8).

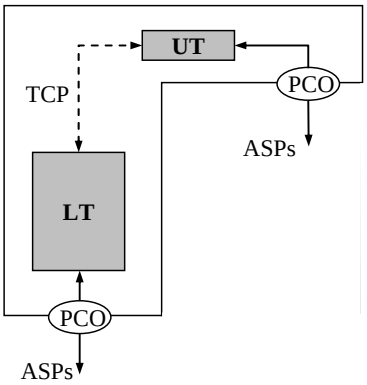


Figura 3.8: Elementos del Método de Pruebas que forman parte del Subsistema de Pruebas.

La estructura detallada de este Subsistema se muestra en la Figura 3.9. Como se observa, está formado:

- Juego de Pruebas Abstractas
- Motor TTCN
- Módulo Adaptador de las Pruebas
- Módulo de Gestión de las Pruebas

El Juego de Pruebas Abstractas modela el comportamiento de las pruebas, es decir, la secuencia de estímulos y respuestas a realizar para verificar un determinado Propósito de Prueba. Este modelado es abstracto y emplea la notación TTCN. La Tabla 3.1 muestra los Juegos de Pruebas Abstractas de distintos sistemas de comunicaciones móviles.

SPICUM servicio de publicaciones

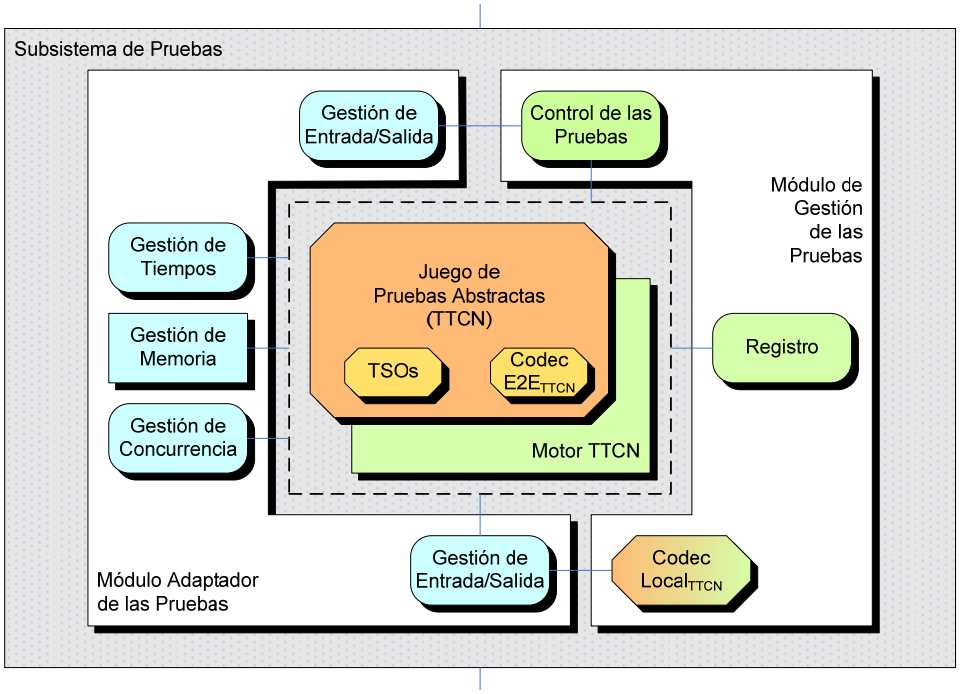


Figura 3.9: Estructura detallada del Subsistema de Pruebas.

Las Pruebas pueden requerir la implementación de dos elementos adicionales: las funciones TSO (Capítulo 2, Sección 2.2.2.4) y un codificador extremo-extremo. El codificador extremo-extremo (Codec $E2E_{TTCN}$) se encarga de codificar, y descodificar, las PDUs que se comunican entre las pruebas y la IUT, ya que TTCN define las primitivas y las unidades de datos de forma abstracta. Puede ocurrir que las constricciones empleadas para asignar valores concretos a las primitivas y unidades de datos estén ya codificadas, pero no será así si esta asignación depende de interacciones previas. Como en el caso de las funciones TSO, TTCN no dispone de construcciones propias para realizar este procesamiento de forma eficiente[3].

Tabla 3.1: Estándares de prueba para diversos sistemas de comunicaciones.

Sistema	Nivel	Estándar	Sistema	Nivel	Estándar
DECT			GSM		
	MAC	[EN 300 497-3] (FT)		DLL	[EN 300 607-1]
	DLC	[EN 300 497-5]		Layer 3	[EN 300 607-3]
	NWK	[EN 300 497-9] (FT)	UMTS		
Bluetooth				MAC	[3GPP 34.123]
	BB	[BTEST BB]		RLC	[3GPP 34.123]
	LM	[BTEST LM]		PDCP	[3GPP 34.123]
	L2CAP	[BTEST L2CAP]		BMC	[3GPP 34.123]
	SPP	[BTEST SPP]		RRC	[3GPP 34.123]
	GAP	[BTEST GAP]		SMS	[3GPP 34.123]
	SDP	[BTEST SDP]		NAS	[3GPP 34.123]

[3] TTCN sí ofrece la posibilidad de indicar el tipo de codificación que requiere una ASP o una PDU. Si se trata de un tipo estándar, como por ejemplo BER o PER, es posible que la herramienta comercial ya disponga de este componente.

SPICUM servicio de publicaciones

El Motor TTCN es el encargado de implementar la semántica de TTCN. Está formado por un conjunto de librerías que se enlazan en tiempo de compilación. Por ejemplo, incluye la funcionalidad para comparar un mensaje recibido con el patrón del evento de recepción, así como la planificación de la sentencia que se debe ejecutar en cada momento.

El Módulo de Gestión de las Pruebas proporciona servicios de alto nivel para la ejecución de los Casos de Prueba, el registro de la ejecución y la comunicación con otros Subsistemas. Los componentes en que se estructura son:

- Control de las pruebas: Ofrece al Subsistema de Operación y Administración una interfaz para la selección de Casos de Prueba, el control de su ejecución y la obtención de veredictos finales y parciales.
- Registro: Procesa, almacena y pone a disposición del Subsistema de Operación y Administración las trazas que se generen durante la ejecución.
- Codec $Local_{TTCN}$: Codificador empleado en la comunicación con el Subsistema Inferior. Es uno de los elementos particulares de cada Sistema de Pruebas pero se puede generar automáticamente (ver Capítulo 5).

El Módulo Adaptador de las Pruebas ofrece acceso a los servicios de bajo nivel de la plataforma de ejecución, típicos de un sistema operativo. Los componentes en que se estructura son:

- Gestión de Entrada/Salida: Para la comunicación con el Subsistema de Operación y Administración (componente superior) y con el Subsistema Inferior (componente inferior).
- Gestión de Tiempos: Ofrece una interfaz de temporizadores. Se pueden activar, cancelar o reiniciar. Cuando un temporizador vence genera un evento.
- Gestión de Memoria: Permite manejar memoria dinámica durante la ejecución de la prueba, según las necesidades del Motor TTCN u otro Módulo.
- Gestión de Concurrencia: Permite la ejecución concurrente de los Componentes de Prueba.

3.2.3 Subsistema Inferior

El Subsistema Inferior es el encargado de comunicar la frontera inferior del Subsistema de Pruebas con el Sistema Bajo Prueba. Realiza una función equivalente a lo que en [ES 201 873-5] se denomina el Adaptador del Sistema Bajo Prueba (*SUT Adaptor*) y corresponde a la parte del Proveedor de Servicio de los Métodos de Pruebas que pertenece al Sistema de Pruebas (Figuras 2.7 a 2.10). La estructura detallada de este Subsistema se muestra en la Figura 3.10 y consta de los siguientes elementos:

- Modelo Abstracto de Protocolos
- Motor SDL
- Módulo Adaptador de los Protocolos
- Módulo de Gestión de los Protocolos
- Módulo de Capa Física

SPICUM servicio de publicaciones

Este Subsistema puede incluir hasta el Nivel Físico, por lo que se ha separado en dos módulos, cada uno de los cuales cubre una problemática de diseño diferente: Módulo de Protocolos y Módulo de Capa Física. El primero está más relacionado con el concepto telemático de protocolo (envío y recepción de mensajes, asincronía, petición-respuesta), mientras que el segundo se refiere más a procesado de señal (modulación, filtros de recepción y transmisión, sincronización temporal, dominio de la frecuencia, …). La separación de funcionalidades entre uno y otro depende de los requisitos de velocidad y sincronización temporal, ya que en el Módulo de Protocolos los tiempos de respuesta pueden ser mayores. Esta separación suele encontrarse en el Nivel del Acceso al Medio, situando las funcionalidades de este nivel en uno u otro módulo según sus características. La comunicación del Subsistema de Pruebas con el Módulo de Capa Física se realiza siempre a través del Módulo de Protocolos.

La funcionalidad del Subsistema Inferior es la de cada uno de los niveles que incorpora de la tecnología bajo prueba, por lo que su complejidad es proporcional a la de dicha tecnología. Adicionalmente, los Casos de Prueba pueden requerir la activación de un comportamiento inválido, como por ejemplo el envío de tramas erróneas, o la medida de parámetros.

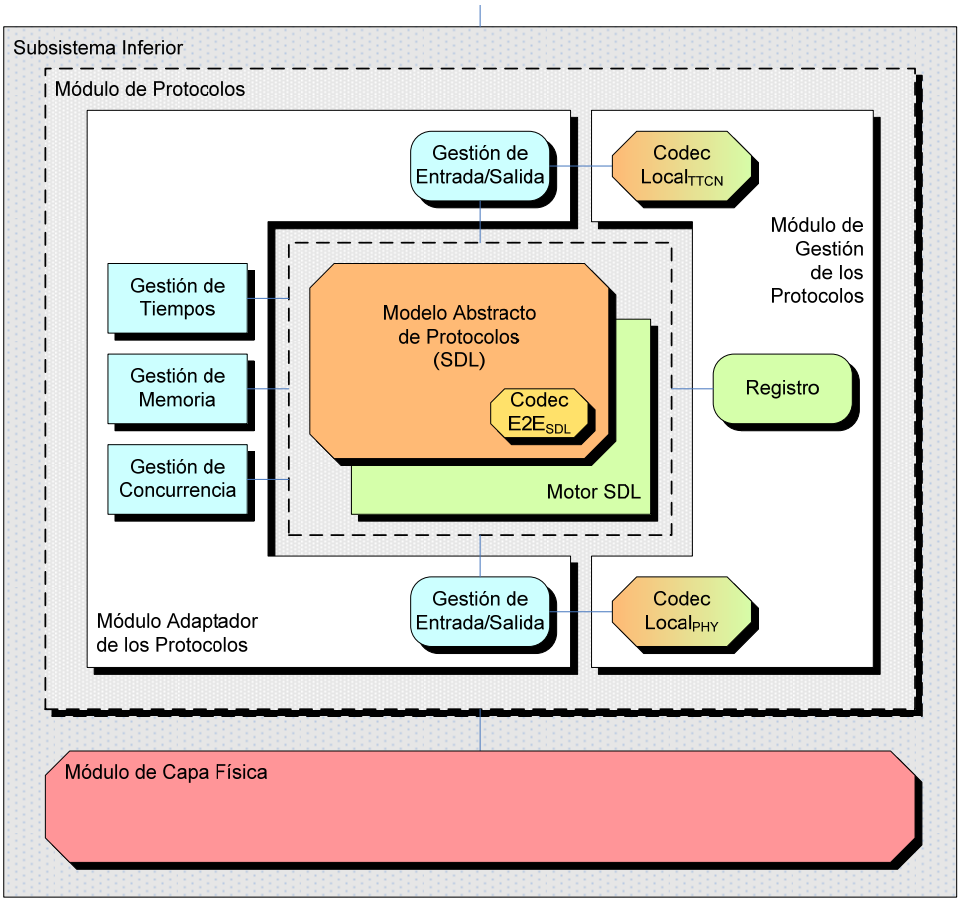


Figura 3.10: Estructura detallada del Subsistema Inferior.

SPICUM servicio de publicaciones

3.2.3.1 Módulo de Protocolos

El Módulo de Protocolos incluye la funcionalidad de los protocolos necesarios para comunicar el Subsistema de Pruebas con el Sistema Bajo Prueba que no exijan una estricta sincronización temporal. Está formado por cuatro módulos.

El Modelo Abstracto de Protocolos representa la máquina de estados que modela el comportamiento abstracto de cada uno de los protocolos. Este Modelo se puede desarrollar en cualquier lenguaje (C, C++, Java, Perl, …) y la arquitectura no restringe esta elección en ningún aspecto. Sin embargo, tanto la arquitectura aquí descrita como el desarrollo de las herramientas de soporte (Capítulo 5) se han organizado en base al principio de diseño de utilizar lenguajes de la familia ITU donde sea posible. En este caso, SDL es la elección evidente, ya que es un lenguaje específicamente pensado para el modelado de sistemas de telecomunicación. Al igual que en el Subsistema de Pruebas, requiere codificadores extremo-extremo (Codec $E2E_{SDL}$) para todas las PDUs intercambiadas con el Sistema Bajo Prueba.

El Motor SDL es el encargado de implementar la semántica de SDL. Está formado por un conjunto de librerías que se enlazan en tiempo de compilación. Por ejemplo, incluye la funcionalidad para seleccionar la siguiente transición a ejecutar, aparentar la ejecución concurrente de los distintos procesos o gestionar las colas de señales.

El Módulo de Gestión de los Protocolos proporciona servicios de codificación y registro de la ejecución. Está organizado en tres componentes:

- Codec $Local_{TTCN}$: Codificador empleado en la comunicación con el Subsistema de Pruebas. Es uno de los elementos particulares de cada Sistema de Pruebas pero se puede generar automáticamente (Capítulo 5).
- Registro: Procesa, almacena y pone a disposición del Subsistema de Operación y Administración las trazas que se generen durante la ejecución.
- Codec $Local_{PHY}$: Codificador empleado en la comunicación con el Módulo de Capa Física. Es uno de los elementos particulares de cada Sistema de Pruebas.

El Módulo Adaptador de los Protocolos ofrece acceso a los servicios de bajo nivel de la plataforma de ejecución, típicos de un sistema operativo. Los componentes en que se estructura son:

- Gestión de Entrada/Salida: Para la comunicación con el Subsistema de Pruebas (componente superior) y con el Módulo de Capa Física (componente inferior).
- Gestión de Tiempos: Ofrece una interfaz de temporizadores. Se pueden activar, cancelar o reiniciar. Cuando un temporizador vence genera un evento.
- Gestión de Memoria: Permite manejar memoria dinámica durante la ejecución de la prueba, según las necesidades del Motor SDL u otro Módulo.
- Gestión de Concurrencia: Permite la ejecución concurrente de las entidades del modelo SDL.

3.2.3.2 Módulo de Capa Física

El Módulo de Capa Física realiza la interfaz física con el Sistema Bajo Prueba. Incluye la funcionalidad más próxima a la interfaz eléctrica del Subsistema Inferior atendiendo a los criterios de velocidad de procesamiento y sincronización temporal. Este módulo es

SPICUM servicio de publicaciones

más afín al área de diseño hardware que al de diseño de protocolos, por lo que no ha sido estudiado en esta Tesis.

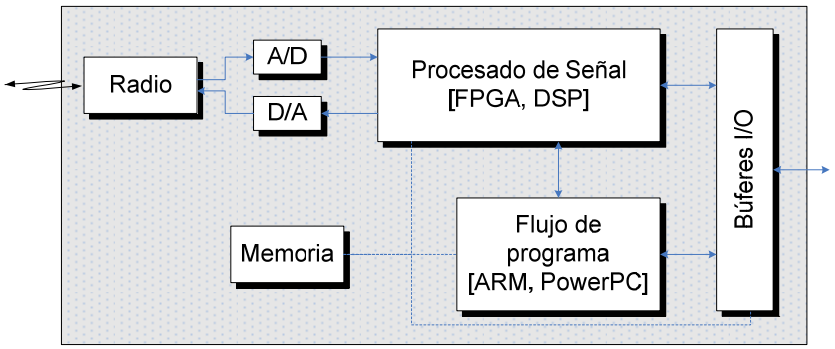


Figura 3.11: Esquema de una estructura flexible para el Módulo de Capa Física.

Como idea general, a partir de la experiencia obtenida, sí se puede sugerir una posible estructura de este módulo, suficientemente flexible como para ser utilizado en la mayoría de Sistemas de Pruebas. Esta estructura se muestra en la Figura 3.11.

El comportamiento del Módulo de Capa Física se separa en dos bloques, en función de la carga computacional requerida: Flujo de Programa y Procesado de Señal. El bloque Flujo de Programa se encarga de ejecutar la máquina de estados de este módulo que, aunque se caracteriza por una estricta temporización, no supone una alta carga computacional. Microprocesadores del tipo ARM o PowerPC son adecuados para realizar este bloque. El bloque de Procesado de Señal implementa la funcionalidad que requiere una mayor carga computacional, tales como moduladores, transformadas de Fourier, filtrado, etc. Para realizar este bloque la mejor alternativa es el uso de FPGAs ([XILINX], [ALTERA]) complementadas con un DSP (*Digital Signal Processor*)[4].

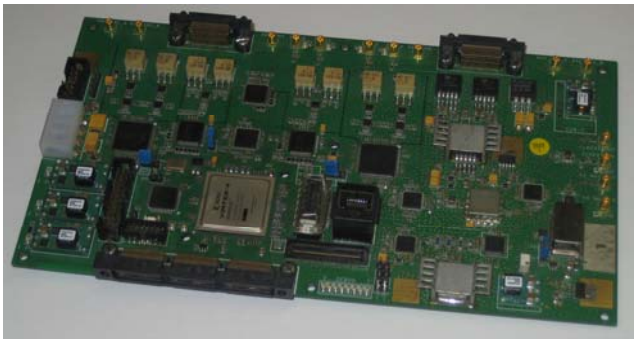


Figura 3.12: Implementación real de un Módulo de Capa Física.

Los búferes de entrada/salida se utilizan para adaptar las distintas velocidades de procesamiento del Módulo de Protocolos y el Módulo de Capa Física. El bloque Radio será un bloque capaz de trabajar en las bandas de frecuencia de la tecnología bajo

[4] La FPGA ofrece la velocidad del hardware pero la versatilidad de ser programable. El uso de un DSP simplifica el diseño, ya que parte de las funciones a implementar se pueden ejecutar en el DSP de que disponga la propia FPGA.

SPICUM servicio de publicaciones

prueba. Los conversores A/D y D/A realizan las conversiones entre señales analógicas y señales digitales.

Un ejemplo de implementación de un Módulo de Capa Física, diseñado para Sistemas de Pruebas construidos con resultados de esta Tesis, se muestra en la Figura 3.12.

3.3 Conclusiones

En este capítulo se ha definido una Arquitectura para Sistemas de Pruebas de equipos de comunicaciones móviles e inalámbricas válida para todo tipo de pruebas. Esta arquitectura define los componentes que forman parte de un Sistema de Pruebas y permite minimizar los costes del desarrollo mediante la formalización de las interfaces y la reutilización de diversos elementos. Así, el esfuerzo de desarrollo de un nuevo Sistema de Pruebas se reduce a la implementación de los componentes específicos y su integración con los componentes genéricos. Para cada uno de los componentes se ha definido su funcionalidad sin limitar posibles implementaciones. La Tabla 3.2 resume cuáles de los componentes han sido implementados como parte de esta Tesis, cuáles se encontraban disponibles previamente y cuáles son específicos de cada Sistema de Pruebas.

La arquitectura se ha diseñado de forma que sea independiente de la plataforma de ejecución, así como de las herramientas comerciales. No obstante, su utilización práctica requiere tener en cuenta las capacidades de los entornos de desarrollo disponibles, lo que ha influido en algunas de las decisiones de diseño. En los capítulos siguientes se presentan una propuesta de Metodología de Diseño a seguir basada en esta arquitectura e implementaciones de los componentes genéricos.

SPICUM servicio de publicaciones

Tabla 3.2: Disponibilidad de los componentes de la arquitectura de un Sistema de Pruebas.

Subsistema	Módulo	Elemento	Disponible	Implementado	Específico
Subsistema de Operación y Administración					
	Interfaz de Operación			X	
	Módulo de Gestión			X	
Subsistema de Pruebas					
	Juego de Pruebas Abstractas				X
		TSOs			X
		Codec $E2E_{TTCN}$			X
	Motor TTCN		X		
	Módulo Adaptador de las Pruebas				
		Gestión de Entrada/Salida		X	
		Gestión de Tiempos		X	
		Gestión de Memoria	X		
		Gestión de Concurrencia		X	
	Módulo de Gestión de las Pruebas				
		Control de las Pruebas		X	
		Registro		X	
		Codec $Local_{TTCN}$		X	X
Subsistema Inferior					
Módulo de Protocolos					
	Modelo Abstracto de Protocolos				X
		Codec $E2E_{SDL}$			X
	Motor SDL		X		
	Módulo Adaptador de los Protocolos				
		Gestión de Entrada/Salida		X	
		Gestión de Tiempos	X		
		Gestión de Memoria	X		
		Gestión de Concurrencia	X		
	Módulo de Gestión de los Protocolos				
		Codec $Local_{TTCN}$		X	X
		Registro		X	
		Codec $Local_{PHY}$		X	X
Módulo de Capa Física					X

SPICUM servicio de publicaciones

Capítulo

4

"'Por tanto, ¿la tarea es su propia justificación?', presionó el Cardenal. '¿Incluso si no tiene un significado más profundo?'
'Tal vez una tarea bien hecha es el significado más profundo', dijo Aenea."

Aenea, La Ascensión de Endymion

CAPÍTULO 4: METODOLOGÍA DE DISEÑO

El diseño es una labor de ingeniería en la que se crea y desarrolla un plan para un sistema, producto, estructura o componente. El proceso de diseño está compuesto por una multitud de actividades, cada una de las cuales se puede ver como un sistema que recibe un conjunto de modelos a su entrada y ofrece un conjunto de modelos a la salida, resultado de aplicar sobre las entradas el procesamiento definido en la actividad (Figura 4.1). El instante en que una actividad se realiza depende no sólo de cuándo están preparadas sus entradas, que, en general, son salidas de otras actividades, sino también de los recursos disponibles, humanos y materiales.

Entre los enfoques típicos para llevar a cabo un diseño se encuentran el diseño centrado en el uso, que se realiza a partir de las metas y tareas asociadas con el uso previsto, y el diseño centrado en el usuario, que pone el énfasis en las necesidades y limitaciones del usuario final.

Cuando la secuencia de actividades necesarias para lograr el objetivo final se estructura y se definen métodos, sistemáticos o no, para realizar cada una de las tareas, el proceso de diseño deja de ser un arte para convertirse en una técnica. Al conjunto integrado de métodos para lograr un propósito se le denomina metodología. La descripción de una metodología marca la línea ideal de seguimiento que se debe buscar; en proyectos reales el diseño avanza en una especie de oscilación alrededor de esta ruta idealizada [ARMI97].

En este capítulo se describe una Metodología de Diseño de Sistemas de Pruebas basada en el uso de lenguajes formales y en la arquitectura presentada en el capítulo anterior ([PONC99], [PONC00]). El enfoque seguido ha sido el de un diseño centrado en el uso. Cada una de las fases está compuesta por una o más actividades, que hacen uso tanto de herramientas comerciales como de herramientas propias, implementadas para cubrir las carencias de las primeras. La Metodología es independiente de herramientas comerciales específicas, pero requiere su uso para una aplicación eficiente de la misma.

SPICUM servicio de publicaciones

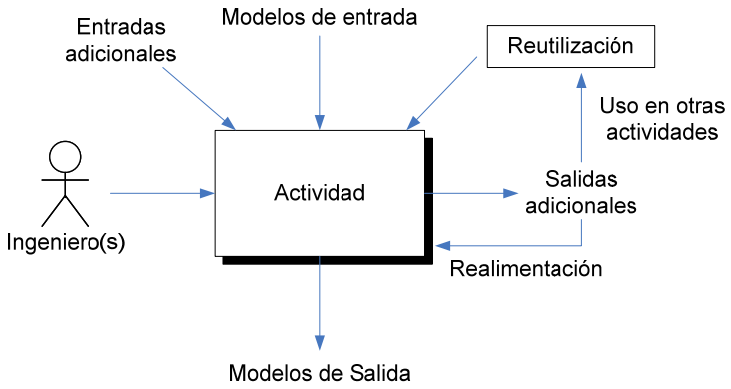


Figura 4.1: Esquema del proceso de ingeniería.

Este capítulo se ha estructurado de la siguiente forma. En primer lugar se describen las ideas generales en que se basa la Metodología de Diseño. Seguidamente, cada fase de la Metodología se describe en una sección separada. Por último, se ofrece una visión conjunta de las actividades que componen la Metodología, junto con una guía para su planificación.

4.1 Visión Global de la Metodología de Diseño

La Metodología de Diseño engloba todas las actividades necesarias para la construcción de un Sistema de Pruebas a partir de las Especificaciones de Sistema y las Especificaciones de Prueba. La Metodología está pensada en función de la Arquitectura de Sistemas de Pruebas descrita en el capítulo anterior. Cada Sistema de Pruebas requiere la implementación e integración tan sólo de aquellos elementos que le son específicos (Figura 4.4); entre ellos, los Juegos de Pruebas Abstractas y el Modelo Abstracto de Protocolos[1]. Es decir, requiere la construcción del Subsistema de Pruebas y del Subsistema Inferior. Las actividades de la Metodología se han organizado de forma que la construcción de ambos Subsistemas pueda realizarse en paralelo, por grupos de trabajo diferentes.

La Figura 4.2 muestra las fases de que consta la Metodología, indicando si su resultado son documentos de texto, modelos o entidades ejecutables; la Figura 4.3 muestra un esquema en forma de árbol de dichas fases. La numeración de cada fase se corresponde con la sección de este capítulo donde se describe dicha fase; esta numeración es la misma en el Capítulo 8 sin más que sustituir el 4 inicial por un 8 (ej: la fase Definición de Interfaces del Módulo de Protocolos se describe en la Sección 4.5.1.2 del Capítulo 4 y corresponde a la Sección 7.5.1.2 del Capítulo 8).

[1] El diseño del Módulo de Capa Física, aunque es un elemento específico para cada tecnología bajo prueba, se encuentra fuera del ámbito de esta Metodología.

SPICUM servicio de publicaciones

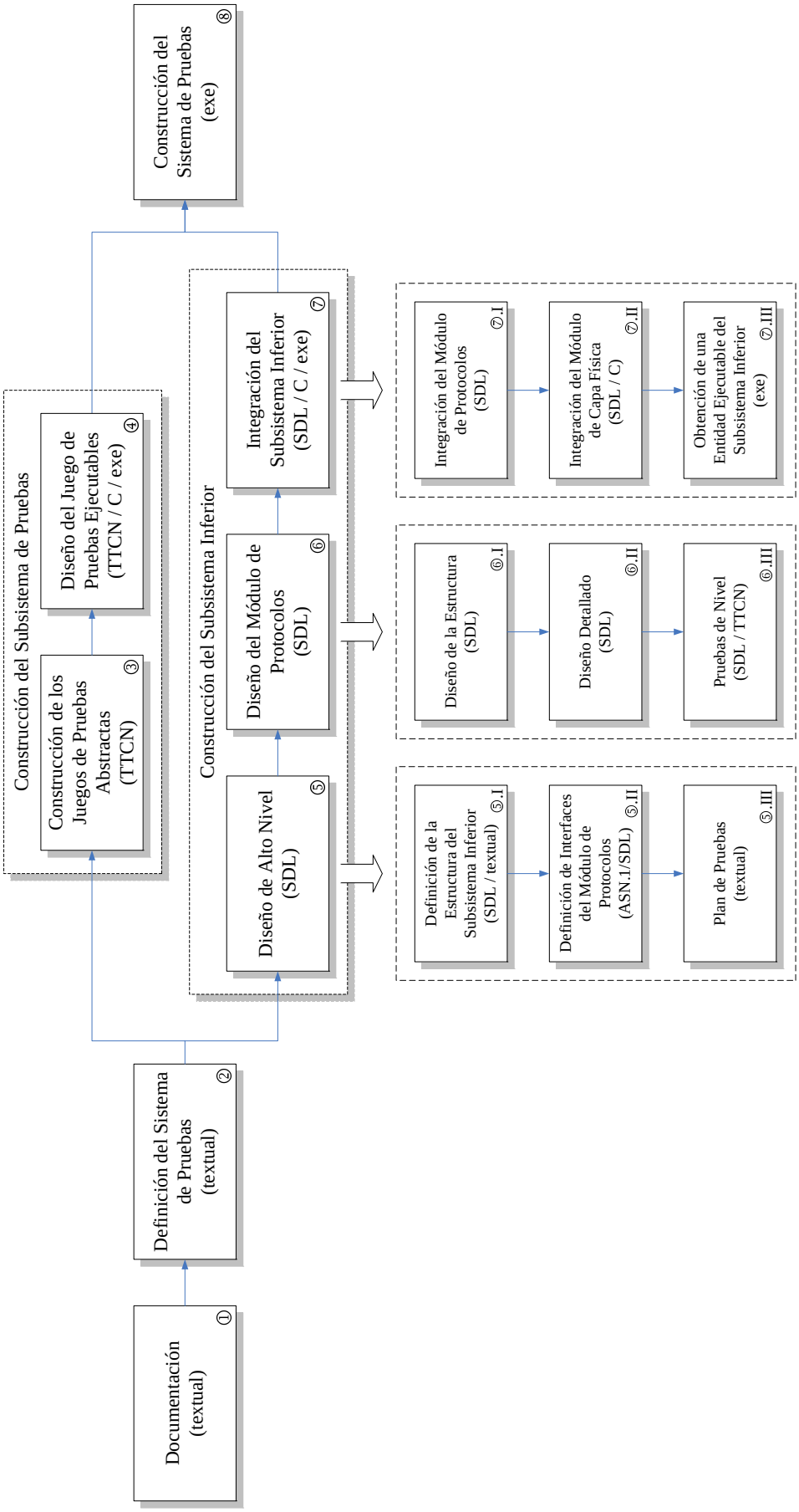


Figura 4.2: Visión global de la Metodología de Diseño.

SPICUM servicio de publicaciones

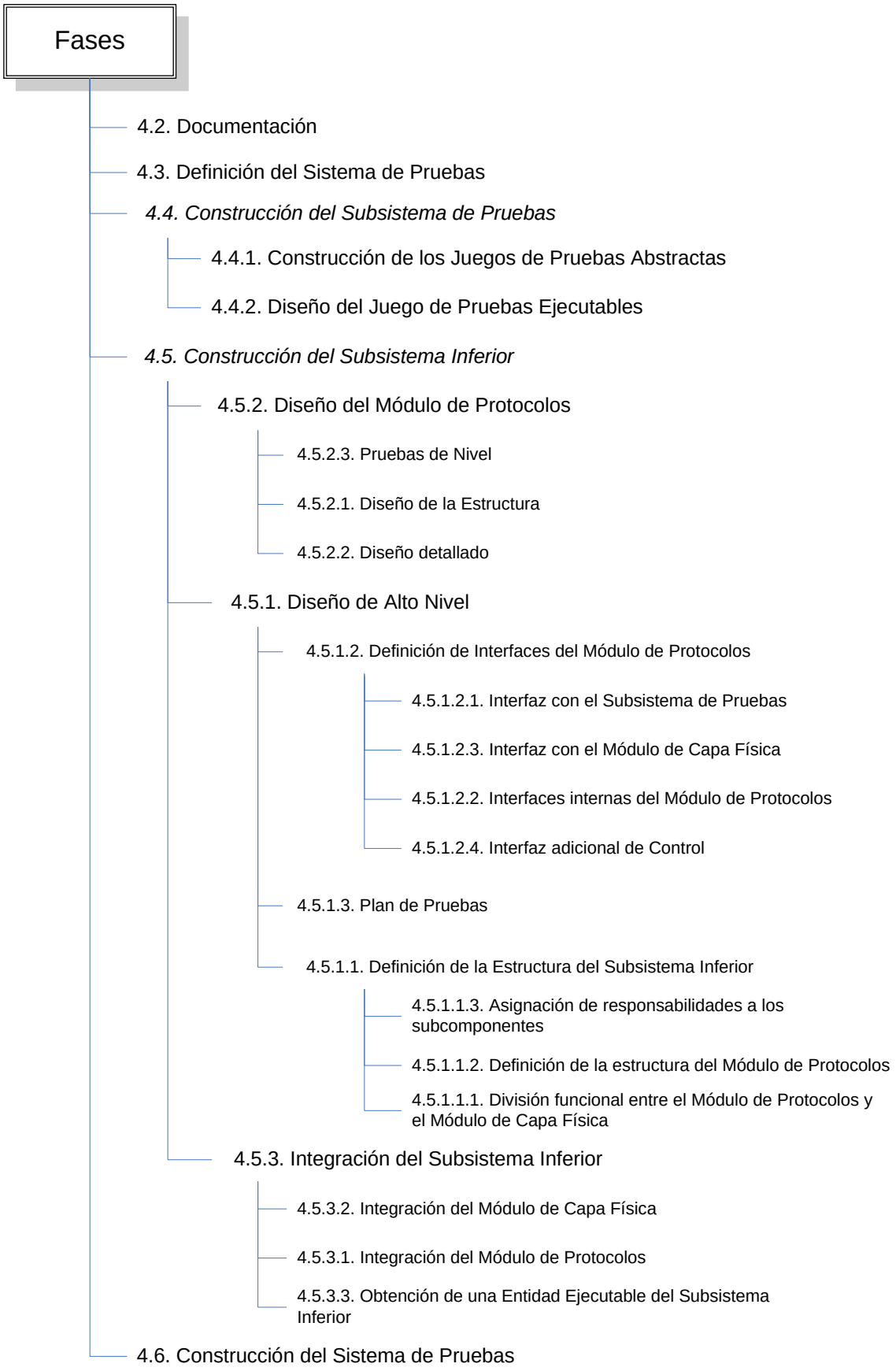


Figura 4.3: Esquema de las fases de la Metodología de Diseño.

SPICUM servicio de publicaciones

Aunque las actividades se presentan en una secuencia lineal, tan sólo se pretende reflejar con esta estructuración las dependencias entre ellas. El diseño de estos Sistemas es un proceso iterativo, donde por lo general interesa disponer con prontitud de un subconjunto de Pruebas en funcionamiento que permita demostrar el Sistema e ir introduciendo el producto en el mercado. La inclusión de nueva funcionalidad o la modificación de la ya existente, por cambios en las especificaciones o por detectar errores en la implementación, requiere retroceder hasta una etapa anterior y realizar un nuevo ciclo de diseño. Tomando una instantánea al final de cada ciclo, el sistema resultante se puede ver como la evolución de los sistemas intermedios logrados al final de cada iteración.

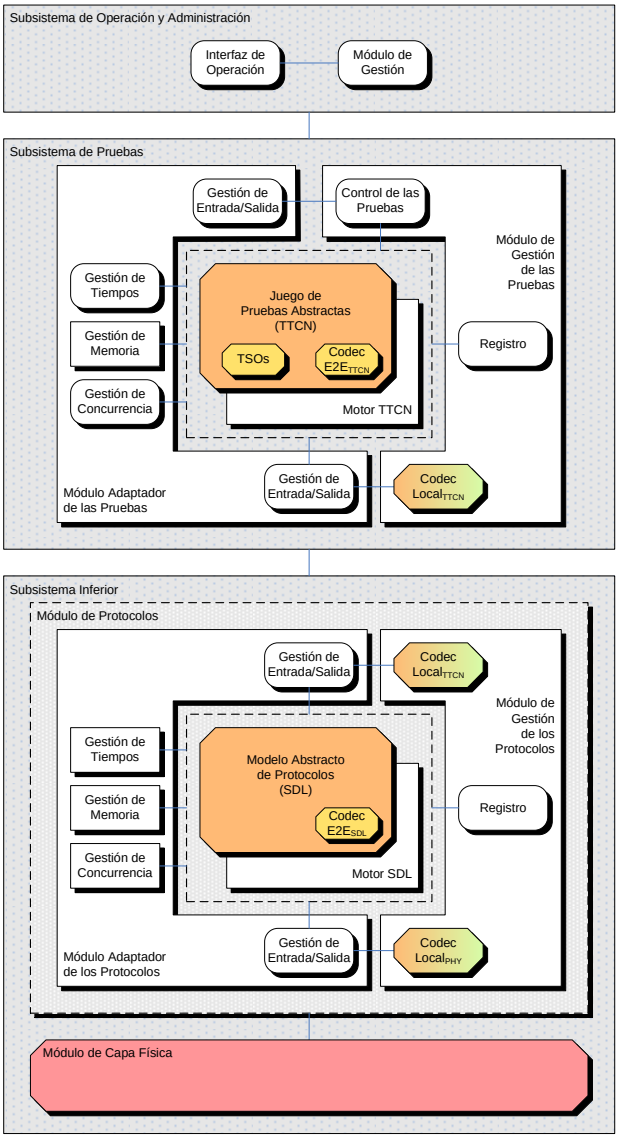


Figura 4.4: Elementos específicos de cada Sistema de Pruebas.

La Metodología asume, de forma natural, el uso de los lenguajes y notaciones de la familia ITU [AMYO99]. Los Juegos de Pruebas se modelan en TTCN; el Módulo de Protocolos se modela en SDL. Para definir interfaces se utiliza la notación ASN.1; tanto TTCN ([X.292], [ES 201 873-1]) como SDL ([Z.105], [Z.107]) aceptan esta notación. La notación MSC se utiliza a lo largo de todo el proceso de diseño como notación auxiliar. Algunos aspectos en los que esta notación resulta de interés son: la identificación de

SPICUM servicio de publicaciones

entidades a alto nivel, la definición de interacciones entre distintas entidades, el análisis de los resultados obtenidos en las pruebas del desarrollo o la visualización gráfica de flujos de mensajes.

En este capítulo no se especifica una metodología concreta para el diseño de sistemas SDL. Lo que sí se indica en este capítulo es una secuencia de actividades que permiten estructurar el diseño de estos sistemas y definir hitos, las salidas de cada fase, en el proceso. La utilización de una metodología de diseño con SDL se ha analizado en el Capítulo 7.

La Metodología de Diseño hace uso de herramientas comerciales y herramientas propias para llevar a cabo las distintas actividades que propone. Se ha asumido que los entornos de desarrollo comerciales (SDL y TTCN) disponen, al menos, de funcionalidades de edición, simulación y generación de código. No obstante, la Metodología es independiente de entornos de desarrollo concretos.

Para aquellas tareas para las que los entornos comerciales no ofrecen una herramienta adecuada, se han diseñado herramientas propias. Estas herramientas, descritas en el Capítulo 5, se agrupan en componentes de la arquitectura y generadores automáticos. Los componentes son aquellos elementos genéricos que se pueden reutilizar en distintos Sistemas de Pruebas; hablamos del Subsistema de Operación y Administración, y los Módulos Adaptador y de Gestión de los Protocolos y las Pruebas. Los generadores permiten extraer información o producir otros componentes de la arquitectura de forma automática.

Dentro de las herramientas propias se ha incluido la especificación de unas Reglas de Nombrado, que formalizan la asignación de nombres a las distintas entidades que se definen un sistema SDL. Con ello se facilita la identificación rápida del elemento al que se refieren y se evita la duplicación de nombres al repartir las labores de diseño entre distintos ingenieros. Estas Reglas de Nombrado no se indican explícitamente como entrada de ninguna fase, pero se deben entender presentes en todas las actividades de diseño en SDL.

La Metodología es también independiente tanto de la plataforma de desarrollo como de la plataforma de ejecución del Sistema de Pruebas resultante. Como plataformas de desarrollo se han utilizado Unix y Windows[2]. Los Sistemas de Pruebas descritos en los Capítulos 8 al 10 se han ejecutado sobre plataformas Linux, Unix, Windows y DSP/BIOS.

El punto de partida de la Metodología de diseño es el conjunto de documentos textuales que especifican el sistema de comunicaciones y sus pruebas; de identificar estos documentos se encarga la fase Documentación. A partir de ellos se procede con la Definición del Sistema de Pruebas, que especifica la funcionalidad que éste debe cumplir. Se continúa con las fases Construcción del Subsistema de Pruebas y Construcción del Subsistema Inferior; estas fases se han subdividido en otras de menor nivel, como se verá a lo largo de este capítulo. En el caso de la construcción del Subsistema Inferior esta división se muestra en la parte inferior de la Figura 4.2.

En el caso de Sistemas de Pruebas de Conformidad de Protocolos los Juegos de Pruebas Abstractas pueden estar disponibles, pero esto no es así para otros casos como Sistemas de Pruebas Radio o Sistemas de Pruebas de Interoperatividad; por ello, una de las actividades es el modelado de dichos Juegos de Pruebas Abstractas. El desarrollo del

[2] El uso de estas plataformas de desarrollo se ha debido a los requisitos de los entornos comerciales.

SPICUM servicio de publicaciones

Módulo de Protocolos, parte del Subsistema Inferior, se realiza de forma incremental. Finalmente, se procede a la integración del Sistema de Pruebas y a su validación.

Aunque durante la explicación de la Metodología puede dar la sensación de que se está trabajando con un único Juego de Pruebas, y, por tanto, un único Módulo de Protocolos, esto es así simplemente por la elección que se ha hecho en el uso del lenguaje para facilitar la descripción. Tener en cuenta el uso de más de un Juego de Pruebas sería como repetir la secuencia de actividades para cada uno de ellos, aunque los Subsistemas Inferiores de cada uno de ellos tendrían un cierto grado de similitud.

Para facilitar la lectura de este capítulo, al inicio de cada sección se incluye una figura reducida que resalta la fase que corresponde en la secuencia de actividades; este gráfico es una visión de alto nivel de las fases de la Metodología de Diseño (Figura 4.5). Al final del apartado dedicado a cada fase se incluye una tabla que muestra las entradas y salidas de la misma. Algunas entradas y salidas de algunas fases son opcionales, ya que su uso u obtención depende de decisiones internas de la propia fase; estas entradas y salidas se denotan en la tabla con la marca '(opc)'. La tabla incluye en la parte inferior las herramientas empleadas; las herramientas comerciales se muestran en la columna de la izquierda mientras que las herramientas propias aparecen en la columna de la derecha.

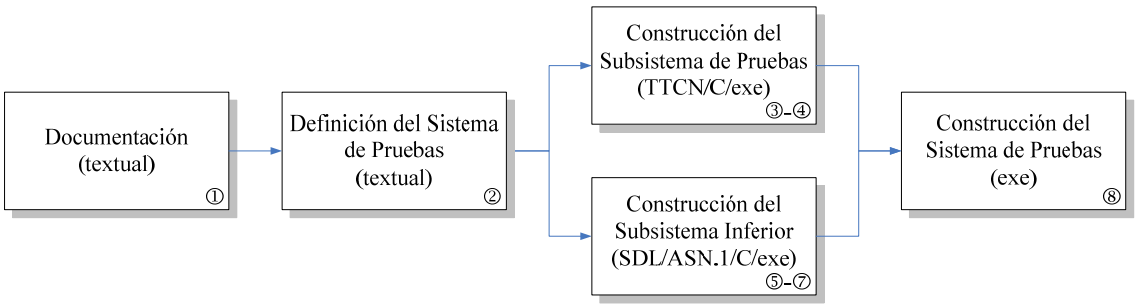


Figura 4.5: Visión de alto nivel de las fases de la Metodología de Diseño.

4.2 Documentación

El punto de partida en el desarrollo de un Sistema de Pruebas es la recopilación de toda la documentación necesaria. Esta documentación se suele encontrar dispersa en varios documentos diferentes. Las razones para ello son fundamentalmente dos: por un lado, una división conceptual, mediante la agrupación de temas afines dentro de cada documento, que ofrece una modularización que simplifica el manejo de la información; por otro lado, una división temporal, que nace del propio proceso de desarrollo de una especificación, generando cada documento en un estadio distinto.

El objetivo de esta fase es obtener una comprensión del sistema de comunicaciones a probar y de los requisitos necesarios del Sistema de Pruebas. Al final se debe disponer de una lista de documentos en los cuales basar el desarrollo.

SPICUM servicio de publicaciones

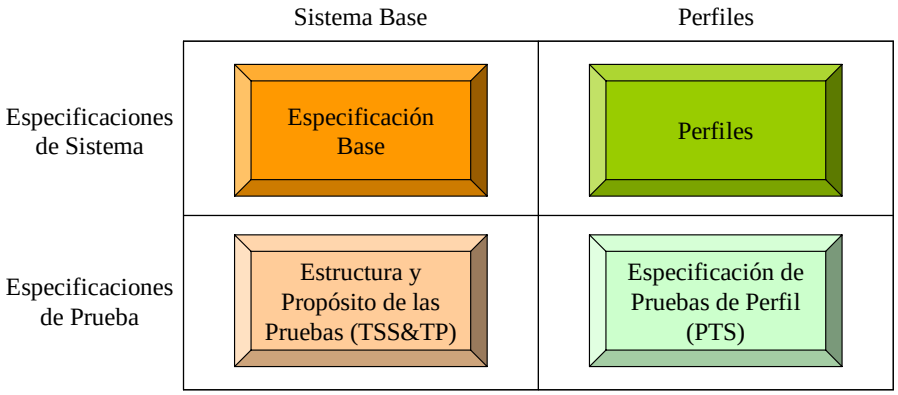


Figura 4.6: Clasificación de las normas de interés en la fase de Documentación.

Las normas[3] se pueden clasificar según el objetivo fundamental de las mismas en Especificaciones de Sistema y Especificaciones de Prueba (ver Figura 4.6). Las Especificaciones de Sistema proporcionan al diseñador la descripción del sistema objeto de interés. Estas normas incluyen las normas que proporcionan la descripción completa del sistema de comunicaciones o Especificación Base y las normas que contienen los Perfiles. La Especificación Base puede estar constituida por uno o más documentos, cada uno de ellos relativo a distintos aspectos del sistema; habitualmente se describe cada nivel del sistema en un documento distinto por modularidad.

Un perfil es la definición de un "*conjunto consistente de opciones de las especificaciones de base para proporcionar una cierta funcionalidad en un entorno dado*" [ETS 300 406]. La especificación de un Perfil se realiza particularizando las opciones que permite la Especificación Base. Por ejemplo (ver Figura 4.7), la Especificación Base del Sistema DECT está descrita en las partes 1 a 8 de la norma [ETS 300 175]; el Perfil de Acceso Genérico (GAP) viene descrito en la norma [ETS 300 444] y especifica la funcionalidad mínima necesaria para la transmisión de voz a 3,1 KHz.

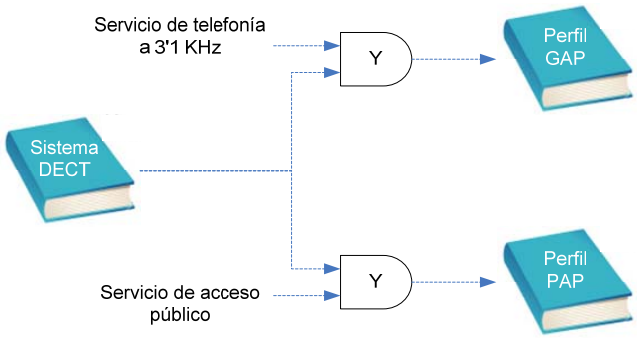


Figura 4.7: Particularización de las Especificaciones de Sistema base de DECT para definir los perfiles GAP y PAP.

Las Especificaciones de Prueba engloban los documentos necesarios para definir las pruebas a realizar sobre la Implementación Bajo Prueba. La información aquí presente

[3] La metodología no hace distinción entre los sistemas estandarizados por organismos oficiales y aquellos diseñados por entidades privadas. En el segundo caso, no sería correcto denominar 'normas' a las Especificaciones de Sistema y de Prueba. A pesar de ello, utilizaremos la misma terminología para ambos casos.

SPICUM servicio de publicaciones

se refiere tanto a las pruebas en sí como a los Métodos de Pruebas, los objetivos perseguidos, etc. Los documentos empleados por la Metodología OSI sobre pruebas de conformidad han sido descritos en el Capítulo 2, Sección 2.2.5. En el caso de existir pruebas de conformidad para un Perfil, la información se encuentra reflejada en la Especificación de Pruebas del Perfil (PTS), que describe la estructura y los propósitos de las pruebas aplicables.

4.2.1 Entradas y Salidas

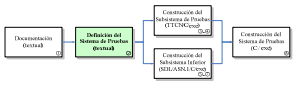
Las entradas de esta fase son las Especificaciones de Sistema y de Prueba y como salida se genera la lista de documentos de interés.

Tabla 4.1: Entradas y salidas de la fase Documentación.

Entradas	Salidas

• Especificaciones de Sistema
  o Especificación Base
  o Perfiles
• Especificaciones de Prueba
  o Estructura y Propósitos de las Pruebas (TSS&TP)
  o Especificación de Pruebas de Perfil (PTS)
• Metodología de Pruebas de Conformidad OSI ([X.290] – [X.296])
	• Lista de Documentos del Sistema

Herramientas	
• Editor de textos	



4.3 Definición del Sistema de Pruebas

En la fase Definición del Sistema de Pruebas se selecciona la funcionalidad que implementará el Sistema de Pruebas, la cual se describe textualmente, no de manera formal. Esta fase parte de un estado en el cual se ha definido el conjunto de pruebas que una implementación debe satisfacer. La principal decisión a la que se enfrenta en este punto el suministrador del Sistema de Pruebas es determinar cuál es su mercado; en concreto, a qué tipo de implementaciones va a dirigir su producto. El suministrador puede decidir que el Sistema de Pruebas sea capaz de verificar la conformidad de cualquier implementación (en cuyo caso deberá incluir todas las pruebas definidas) o puede optar por restringir la funcionalidad del Sistema de Pruebas a un determinado conjunto de implementaciones o a funcionalidades concretas de las mismas (en este caso tan sólo será necesaria la inclusión de aquellas pruebas relevantes para estas implementaciones o estas funcionalidades).

Por ejemplo, un Sistema de Pruebas de equipos DECT podría:

a) Disponer de la capacidad necesaria para certificar cualquier equipo DECT;

b) Estar restringido a la certificación del perfil GAP;

SPICUM servicio de publicaciones

c) Incluir sólo las pruebas correspondientes a una de los niveles del sistema DECT (ej: MAC, DLC, NWK);

d) Ser capaz de verificar el establecimiento de una conexión de voz.

Si se incluyen en esta fase condicionantes derivados de la puesta en el mercado del Sistema de Pruebas, queda claro que no todos los subconjuntos posibles de pruebas tienen sentido desde un punto de vista económico. De los ejemplos puestos anteriormente, tan sólo los tres primeros podrían entenderse como viables financieramente[4].

Desde este mismo punto de vista, no resulta rentable enfocar la construcción del Sistema de Pruebas como un 'todo o nada'. Con objeto de obtener retornos financieros lo antes posible, se requiere un plan de negocio que prevea la obtención de versiones del Sistema de Pruebas con una funcionalidad reducida, lo que conlleva la clasificación de los Casos de Prueba según su importancia estimada para los posibles clientes. Este enfoque permite realizar demostraciones del producto a corto-medio plazo, aunque la implementación de todas las pruebas se alargue durante 1 ó 2 años más. Al mismo tiempo, se facilita la recogida de opiniones de dichos clientes sobre el Sistema de Pruebas, lo que puede aconsejar modificaciones en el mismo o en su funcionalidad.

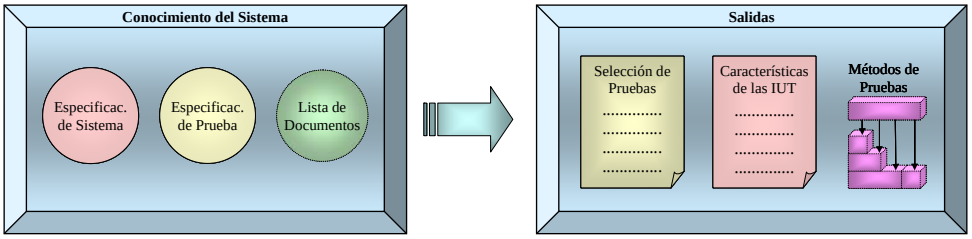


Figura 4.8: Visualización gráfica de las entradas y salidas de la fase Definición del Sistema de Pruebas.

La Metodología de Diseño es independiente del hecho de que se seleccione un mayor o menor conjunto de Pruebas; este factor sólo redundará en el esfuerzo necesario para disponer de la versión final del Sistema de Pruebas. La elección del Subconjunto de Pruebas lleva asociada la determinación del Método de Pruebas. En caso de que una Prueba pueda ser realizada mediante más de un Método de Pruebas, hay que decidir cuál de ellos se va a utilizar (Capítulo 2, Sección 2.2.3). Si este caso no se da, la elección de los Casos de Prueba a realizar determina unívocamente el (o los) Método(s) de Pruebas.

4.3.1 Entradas y Salidas

Las entradas de esta fase no sólo son técnicas; la decisión de qué (o en qué orden) funcionalidad vaya a implementarse depende también de criterios empresariales, aunque no quedan reflejados en la lista de entradas. Las salidas indican qué Pruebas se van a implementar y qué IUTs son susceptibles de ser probadas (Figura 4.8).

[4] Es previsible que un Sistema de Pruebas que tan sólo disponga de las Pruebas para verificar el establecimiento de una conexión de voz carezca de mercado ya que no es capaz de certificar ninguna funcionalidad lógica completa

SPICUM servicio de publicaciones

Tabla 4.2: Entradas y salidas de la fase Definición del Sistema de Pruebas.

Entradas	Salidas
• Lista de Documentos del Sistema • Especificaciones de Sistema • Especificaciones de Prueba	• Lista de Casos de Prueba a implementar • Métodos de Pruebas requeridos • Características del segmento de implementaciones a probar
Herramientas	
• Editor de textos	

4.4 Construcción del Subsistema de Pruebas

El Subsistema de Pruebas es el responsable de realizar el comportamiento de las Pruebas. El conjunto de etapas que se ha denominado 'Construcción del Subsistema de Pruebas' integra todas las actividades necesarias para la obtención de un ejecutable de dicho componente de la arquitectura. Para su generación se requiere seleccionar el conjunto de pruebas que se va a implementar e integrarlas con los Módulos de Gestión y Adaptación de las Pruebas. Bajo el criterio de utilizar lenguajes de la familia ITU, se ha considerado que los Juegos de Prueba, disponibles o no, se modelan en TTCN. Los elementos que hay que realizar para obtener el Subsistema de Pruebas se muestran en la Figura 4.9.

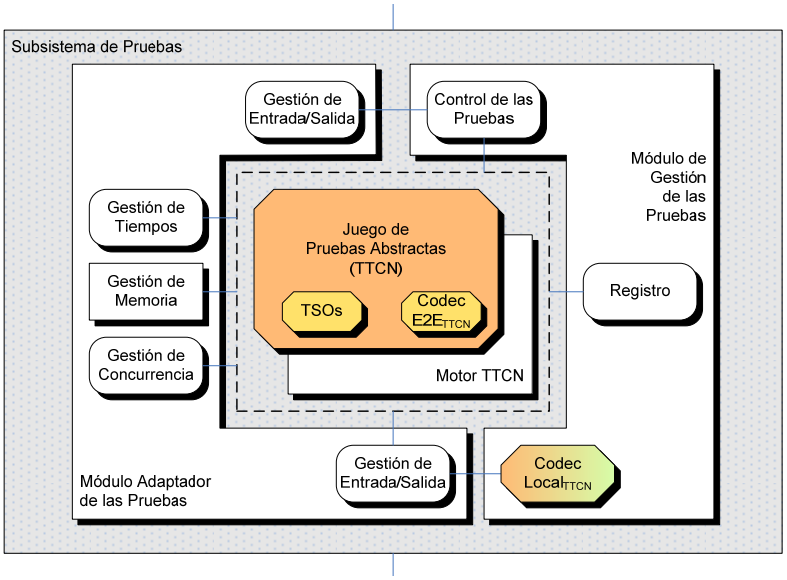


Figura 4.9: Elementos que hay que realizar en la Construcción del Subsistema de Pruebas.

La construcción del Subsistema de Pruebas se ha desglosado en dos fases (Figura 4.10):

1. Construcción de los Juegos de Pruebas Abstractas: Tiene como objetivo el modelado y validación de los distintos Casos de Prueba.
2. Diseño del Juego de Pruebas Ejecutables: En esta fase se genera un ejecutable de las Pruebas que ofrecerá el Sistema de Pruebas. Para ello es necesario

SPICUM servicio de publicaciones

implementar algunos elementos adicionales (TSOs y codificadores) y enlazarlo con el resto de los componentes mostrados en la Figura 4.9 (Motor TTCN y Módulos Adaptador y de Gestión de las Pruebas).

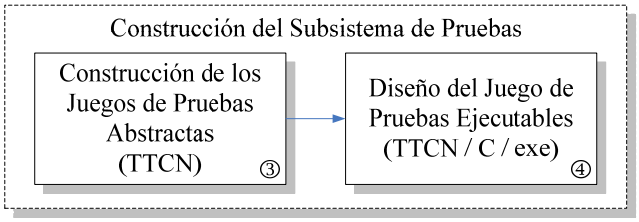


Figura 4.10: Fases para la construcción del Subsistema de Pruebas.

4.4.1 Construcción de los Juegos de Pruebas Abstractas

Los Juegos de Pruebas Abstractas (ATS) suelen desarrollarse bajo la responsabilidad de los mismos organismos (de estandarización o privados) que se han encargado de las especificaciones del sistema. Este desarrollo puede hacerse de forma manual o mediante métodos de generación automática. En el primer grupo se encuentran, por ejemplo, los Juegos de Pruebas diseñados por ETSI para sistemas como DECT o GSM, que fueron de los primeros sistemas en incorporar la Metodología de Pruebas de Conformidad, sistemas patrocinados por la industria como Bluetooth e incluso sistemas privados con vistas a ofrecer un servicio público[5]. La generación automática ([DAHB90], [POST94], [QIXI96], [DSSO99], [BOUR01], [PRAS05], [LEEA06], [KOVA06]) es una técnica cuyo uso se está extendiendo, sobre todo en el ámbito de los organismos de estandarización, pero aún requiere un guiado manual[6].

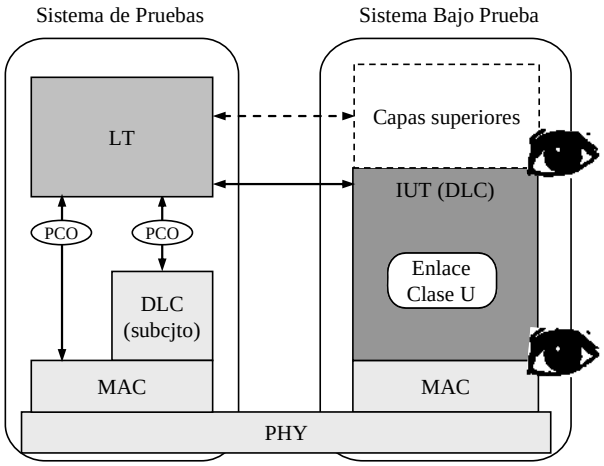


Figura 4.11: Fronteras observables en las pruebas del Nivel DLC de DECT.

[5] Por ejemplo, el sistema de comunicaciones por satélite ICO ([ICO00], [MORA99]).

[6] El mayor problema es truncar el conjunto de Casos de Prueba obtenidos de forma que el proceso de certificación sea viable. ETSI [MTS] afirma que, aunque las herramientas actuales no ofrecen una utilidad completamente automatizada, sí permiten reducir el esfuerzo y el tiempo del desarrollo ([EK97], [KERB99], SCHM00], [GRAB97]).

SPICUM servicio de publicaciones

El modelado de un Juego de Pruebas se basa en los Propósitos de Prueba (TP) y el (o los) Método de Pruebas elegido. Esta información se encuentra recogida en el documento Estructura y Propósito de las Pruebas (TSS&TP)[7]. El Propósito de Prueba se modela teniendo en cuenta la arquitectura definida por el Método de Pruebas mediante un Caso de Prueba. Siguiendo la Metodología de Pruebas de Conformidad, los Juegos de Pruebas se escriben en notación TTCN. Cada nivel del sistema de comunicaciones suele disponer de un Juego de Pruebas propio; por ejemplo, en el caso del sistema DECT existen Juegos de Pruebas para los niveles MAC, DLC y NWK (ver Apéndice C, Sección C.1).

El Método de Pruebas Abstractas (ATM) ha de ser uno de los definidos en la Metodología de Pruebas de Conformidad de OSI (Capítulo 2, Sección 2.2.3). Aunque un mismo Caso de Prueba puede realizarse mediante varios Métodos de Pruebas diferentes, puede ocurrir que las interacciones permitidas no posibiliten la comprobación del Propósito de la Prueba. Por ejemplo, uno de los Propósitos de Prueba (TPUV-000) definidos para el nivel DLC del sistema DECT [EN 300 497-4] requiere comprobar que la IUT, al recibir una cierta trama, considere que se ha establecido un enlace clase U. Como se observa en el Método de Pruebas correspondiente al nivel DLC (Figura 4.11), la interacción con el Proveedor de Servicio se realiza a través de dos Puntos de Control y Observación[8]. La implementación de este Propósito de Prueba no es viable, ya que esta decisión es un estado interno del nivel DLC en la IUT. Con este Método de Pruebas no es posible observar este estado, pues sólo se permite el conocimiento de la información transmitida a través de las fronteras (superior o inferior) del nivel DLC. El Juego de Pruebas debe incluir una lista de los Propósitos de Prueba no verificables debido a limitaciones propias de los Métodos de Pruebas disponibles. En el ejemplo comentado, la imposibilidad de verificación se encuentra documentada en el Juego de Pruebas [EN 300 497-5].

Figura 4.12: Etapas en la construcción de los Juegos de Pruebas.

El proceso de elaboración de los Juegos de Pruebas es una tarea larga y costosa que se puede desglosar en varias etapas (ver Figura 4.12):

- ✓ **Modelado**: El primer paso es llevar a cabo el modelado de las pruebas. El modelado implica dos aspectos[9]:

[7] El documento TSS&TP puede venir auxiliado de diagramas de secuencia de mensajes para ilustrar el comportamiento esperado de cada prueba.

[8] Como se ha visto en el Capítulo 2, son interfaces que permiten observar e influir el comportamiento de la IUT

[9] ETSI pone a disposición de los diseñadores una serie de documentos que sirven de guía para la escritura tanto de los Juegos de Pruebas Abstractas como de los informes y formularios. Así, es posible obtener un resultado de mayor calidad y más fácilmente legible por todas las personas implicadas en los procesos de diseño, desarrollo y certificación. El informe técnico [ETR 141] ofrece un conjunto de recomendaciones a seguir a la hora de escribir un Juego de Pruebas.

SPICUM servicio de publicaciones

a) Escritura de los Juegos de Pruebas: Se modelan en TTCN. En el caso de trabajar con Perfiles, el documento correspondiente se denomina Especificación de Pruebas Específicas del Perfil (PSTS).

b) Definición del formato y contenido de cuantos informes adicionales sean necesarios: Atañen tanto a suministradores como a laboratorios de pruebas. Ejemplos de parámetros incluidos en estos formularios se muestran en la Tabla 4.3.

✓ **Validación**: Esta tarea debe llevarse a cabo por personas diferentes de aquellas que han modelado los Juegos de Pruebas, pero que al mismo tiempo tengan un conocimiento suficientemente amplio tanto del sistema de comunicaciones como de la Metodología de Pruebas de Conformidad. A raíz de los resultados de esta validación puede ser necesario realizar modificaciones en los Juegos de Prueba. El informe [ETR 141] puede servir de base a la hora de llevar a cabo esta validación. La semántica de correctitud de un Juego de Pruebas se comenta en [BÄR92]. Si no se han encontrado errores graves que obliguen a un nuevo ciclo del proceso, los Juegos de Pruebas pueden considerarse aceptados[10].

Tabla 4.3: Ejemplos de parámetros PICS y PIXIT de las pruebas de conformidad de UMTS.

	Nombre	Descripción
PICS	pc_CS	Soporta conmutación de circuitos.
	pc_DetachOnPwrDn	Soporta desconexión de la red al apagarse.
	pc_SMS_CellBroadcast	Capacidad para recibir y mostrar mensajes de difusión.
	pc_SupportOpModeA	TRUE si el UE ofrece simultáneamente servicios de conmutación de circuitos y conmutación de paquetes.
PIXIT	px_CipheringAlgorithm	Algoritmo de cifrado.
	px_CN_DomainTested	Dominio de red para las pruebas.
	px_IMEI_Def	Identificación del suministrador del equipo (IMEI) por defecto.
	px_PriScrmCodeE	Código primario de entrelazado para la celda E.
	px_PTMSI_2	Segundo P-TMSI[11] usado para pruebas.
	px_UE_OpModeDef	Modo de operación del UE por defecto.

✓ **Mantenimiento**: Es infrecuente que no se encuentren errores durante la utilización de un Juego de Pruebas. Por ello, es necesaria su revisión basándose en los comentarios recibidos tanto de los suministradores de Sistemas de Pruebas como de los laboratorios de pruebas que empleen estos equipos. Las sugerencias emitidas por estos últimos son de un alto valor ya que son las organizaciones con un contacto más próximo a los suministradores de equipos, encargados de certificar su correcto funcionamiento, a la vez que por sus manos circula una gran variedad de implementaciones. A partir de estos informes de erratas, que en principio no deberían afectar a funciones esenciales gracias a la validación realizada, se generan nuevas versiones de los Juegos de Pruebas. En pocos años, las posibles ambigüedades en la norma se pueden solventar, incluyendo modificaciones en la misma.

[10] En el caso de un organismo estandarizador se requiere su aprobación mediante los mecanismos habituales de adopción de estándares.

[11] P-TMSI – *Packet TMSI*; TMSI – *Temporary Mobile Subscriber Identity*.

SPICUM servicio de publicaciones

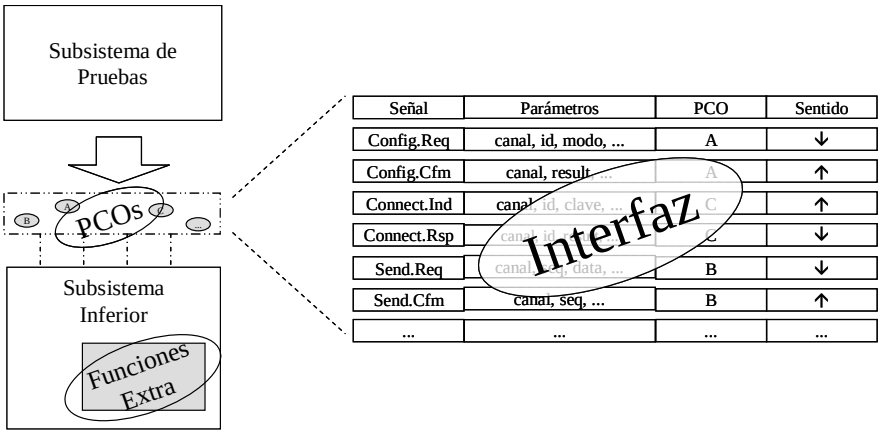


Figura 4.13: Elementos de los Juegos de Pruebas que permiten avanzar en el desarrollo del Sistema de Pruebas.

Es normal que las Especificaciones de Sistema se publiquen sin estar disponibles los correspondientes Juegos de Pruebas, ya que la generación de los Juegos de Pruebas lleva tiempo. Si se esperara a disponer de los Juegos de Pruebas completos antes de iniciar la construcción del Sistema de Pruebas, éste se lanzaría al mercado con un considerable retraso. Esto se puede evitar si se definen tempranamente aquellos elementos que tienen influencia en el diseño del resto de los componentes (Figura 4.13):

- Puntos de Control y Observación.
- Primitivas, incluyendo su semántica y parámetros.
- Funcionalidad adicional que debe ofrecer el proveedor de servicio.

4.4.1.1 Entradas y Salidas

Las entradas de esta etapa son los documentos identificados en la fase Documentación. El resultado no es sólo el conjunto de Juegos de Pruebas necesarios, sino también aquellos formularios que deben ser rellenados por el suministrador o por el laboratorio de pruebas.

SPICUM servicio de publicaciones

Tabla 4.4: Entradas y salidas de la fase Construcción de los Juegos de Pruebas Abstractas.

Entradas	Salidas
• Lista de Documentos del Sistema • Notación TTCN ([X.292] , [ES 201 873-1]) • Notación ASN.1 ([X.680], [Z.105])]	• Juegos de Pruebas Abstractas (ATS) • Especificación de Pruebas Específicas de Perfil (PSTS) • Formularios o Declaración de Conformidad de la Implementación (PICS) o Información Suplementaria para Pruebas (PIXIT) o Declaración de Conformidad del Sistema (SCS) o Informe de Pruebas de Conformidad del Sistema (SCTR) o Informe de Pruebas de Conformidad del Protocolo (PCTR)
Herramientas	
• Entorno de desarrollo en TTCN o Editor o Analizador sintáctico y semántico	

4.4.2 Diseño del Juego de Pruebas Ejecutables

El Subsistema de Pruebas está formado por el Comprobador Inferior (LT) y el Comprobador Superior (UT) (Figura 4.14). Cada uno de estos elementos se encarga de estimular a la IUT y de observar su comportamiento en puntos diferentes. En el caso de que el Juego de Pruebas contenga un comprobador superior, el suministrador de la IUT es el responsable de implementar en el mismo los medios para establecer la comunicación entre ambos. La interfaz a través de la cual se lleva a cabo esta comunicación suele ser distinta de aquellas necesarias para el normal funcionamiento del equipo.

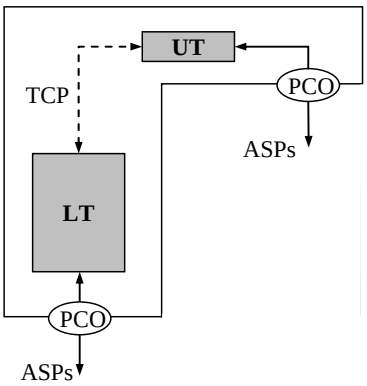


Figura 4.14: Diagrama conceptual del Subsistema de Pruebas.

SPICUM servicio de publicaciones

Los pasos a realizar para obtener un ejecutable del Subsistema de Pruebas son los siguientes[12] (Figura 4.15):

1. Selección de Casos de Prueba: En caso de seleccionar un subconjunto de Casos de Prueba, éste puede extraerse de los Juegos de Pruebas Abstractas mediante alguna de las herramientas comerciales (Capítulo 5, Sección 5.1) que existen. En las fases posteriores, por simplicidad en el lenguaje, no se distinguirá entre la Selección de Casos de Prueba y el Juego de Pruebas, empleando el segundo nombre para referirnos a ambos conceptos.

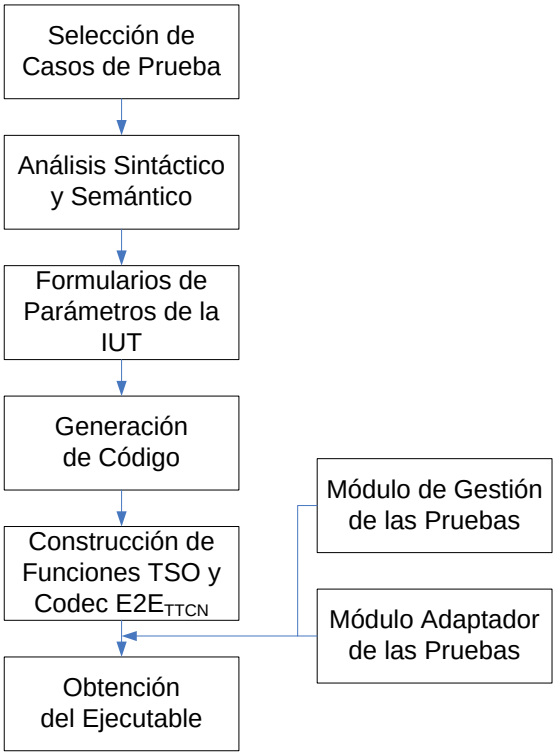


Figura 4.15: Tareas para la generación del ejecutable del Subsistema de Pruebas.

2. Análisis sintáctico y semántico de los Juegos de Pruebas: Aunque la corrección notacional de los Juegos de Pruebas debe haber sido comprobada antes de hacerlos públicos (o de proceder a su entrega) lo cierto es que puede haber errores que hayan pasado inadvertidos. Por ejemplo, las primeras versiones de Juegos de Pruebas de DECT, Bluetooth y UMTS generaban errores a la hora de llevar a cabo un sencillo análisis sintáctico, sin entrar a valorar su corrección respecto a sus Propósitos de Prueba y los estímulos y eventos que empleaban[13].

   Para poder avanzar, a la espera de una solución definitiva de los problemas que puedan haber surgido, tal vez sea necesario modificar ligeramente los Juegos de Prueba, pero sin alterar la semántica de los mismos. Aunque estos retoques se

[12] Salvo indicación en contra, se debe asumir que los pasos a realizar deben llevarse a cabo para cada uno de los Juegos de Pruebas de interés, aunque no quede así reflejado en el texto con objeto de facilitar la lectura del mismo.

[13] Puede que se hayan utilizado para el desarrollo herramientas diferentes de las que se utilicen en esta etapa y que cada una de ellas interprete de forma diferente algunas construcciones del lenguaje, o que estas herramientas posean fallos.

SPICUM servicio de publicaciones

hagan aquí, la versión final del producto debe atenerse en todo caso a la versión oficial de los Juegos de Pruebas.

3. Extracción de Formularios de Parámetros de la IUT: A partir de los Juegos de Pruebas se pueden generar automáticamente los formularios referentes a los Parámetros de Conformidad (PICS) y los Parámetros Adicionales de Implementación (PIXIT). Se ha desarrollado la herramienta *GenDef* para realizar esta tarea (Capítulo 5, Sección 5.5.2.1). Esta herramienta permite también obtener las declaraciones de las funciones TSO requeridas por los Juegos de Prueba, junto con la descripción de su algoritmo.
4. Generación de Código: Las herramientas actuales generan código C, C++ o Java a partir de los módulos TTCN; este código suele estar pensado para poder utilizar compiladores típicos (gcc, Borland C++, MS Visual C++), tanto comerciales como de libre distribución, y poder ejecutarse en la mayoría de plataformas. Se ha decidido emplear código C para utilizar el mismo lenguaje en el código generado a partir de los modelos abstractos del Subsistema de Pruebas y del Subsistema Inferior.
5. Construcción de Componentes:
   a. Funciones TSO: La generación de código se limita a generar las definiciones requeridas por estas funciones para poder ser invocadas, pero el cuerpo de las mismas se encontrará vacío. Es necesario, pues, escribir el código que les permita realizar su objetivo. Estas funciones pueden ir desde simples operaciones hasta complicados algoritmos[14]. Su implementación puede requerir invocar cierto comportamiento de la IUT[15] o del Subsistema Inferior[16]; en este caso, la comunicación se establecerá mediante primitivas que se transmitan a través de un canal específico (ver Sección 4.5.1.2.4).
   b. Codificador Extremo-Extremo (Codec $E2E_{TTCN}$): La comunicación con la IUT requiere la codificación de las PDUs empleadas en el Juego de Pruebas. En la evolución de las herramientas, éstas disponen ya de codificadores habituales (ASCII, BER, PER), y los Juegos de Pruebas ya indican cuál es el que hay que usar. Sin embargo, si es un codificador no disponible, o el Juego de Pruebas no lo especifica, es necesario su implementación.
6. Obtención del Ejecutable: Una vez se dispone de todos los componentes, la última tarea es la compilación y el enlazado de los mismos. Esto se puede realizar con diversos compiladores tanto comerciales como de libre distribución. Para obtener el ejecutable del Subsistema de Pruebas hay que compilar y enlazar los siguientes elementos:
   - Código generado en el paso 4
   - Motor TTCN

[14] Por ejemplo, la función `O_BitstringXOR`, que calcula la operación lógica XOR de dos cadenas de bits en UMTS, o la función `TSO_rfcomm_session`, que realiza una petición SDP e inicia una sesión RFCOMM en Bluetooth.

[15] Cuando no hay Comprobador Superior en el Juego de Pruebas, cualquier operación que se desee realizar sobre la frontera superior de la IUT se llevará a cabo mediante un operador. En este caso, la TSO se limita a solicitar por pantalla esta acción y a obtener una indicación de éxito o no.

[16] Por ejemplo, la función `TSO_jam_traffic_bearer`, que genera interferencias en el canal ocupado para forzar un traspaso intracelda en DECT.

SPICUM servicio de publicaciones

- Componentes construidos en el paso 5: Funciones TSO y Codec $E2E_{TTCN}$
- Módulo Adaptador de las Pruebas
- Módulo de Gestión de las Pruebas

Con anterioridad a la disponibilidad de herramientas comerciales, diversos grupos de investigación, así como algunas empresas, han construido entornos de desarrollo para TTCN. Algunos ejemplos son TOCS [PROB89], FOREST [KATS91], AICTS [KAZA95] y HARPO [ALGA94].

4.4.2.1 Entradas y Salidas

Las entradas de esta fase son los Juegos de Pruebas Abstractas, y las salidas son los componentes necesarios para obtener el ejecutable del Subsistema de Pruebas y los formularios de Parámetros de Pruebas.

Tabla 4.5: Entradas y salidas de la fase Diseño del Juego de Pruebas Ejecutables.

Entradas	Salidas
• Juegos de Pruebas Abstractas • Módulo Adaptador de las Pruebas • Módulo de Gestión de las Pruebas • Motor TTCN	• Subsistema de Pruebas o Ejecutable o Codificación de las funciones TSO o Código generado (C) a partir de los módulos TTCN o (opc) Codec $E2E_{TTCN}$ • Selección de Casos de Pruebas • Formularios o Parámetros de Pruebas (PICS, PIXIT)
Herramientas	
• Entorno de desarrollo en TTCN o Editor o Analizador sintáctico y semántico • Entorno de desarrollo en C o Editor o Compilador	• Generador de Interfaces o Generador de la Definición de la Interfaz

4.5 Construcción del Subsistema Inferior

El Subsistema Inferior está dividido en dos módulos: Módulo de Protocolos y Módulo de Capa Física. En esta Metodología se ha tenido en consideración solamente el Módulo de Protocolos, ya que el Módulo de Capa Física requiere técnicas de diseño hardware, lo cual se encuentra fuera de este ámbito. Para la implementación del Módulo de Protocolos se ha elegido el lenguaje SDL, que permite un modelado abstracto de alto nivel; este modelo puede ser fácilmente adaptado a distintas plataformas. SDL se ha utilizado en distintos ámbitos, como son la especificación, la implementación y el análisis de sistemas: UMTS ([NELS96], [ALON97]), ISDN ([GUOC97]), SS7 ([CROW93]),

SPICUM servicio de publicaciones

ATM ([SIEB97]), WLAN ([RUIZ06], [KIM01]), Gestión OSI ([RODR99]), Inmarsat ([MITC93]), etc.

La construcción del Subsistema Inferior se ha desglosado en tres fases (Figura 4.16). Dada la complejidad de estas fases, se han dividido a su vez en tres actividades, que aparecen en la parte inferior. Las responsabilidades de cada fase son las siguientes:

1. Diseño de Alto Nivel: Su objetivo es definir la estructura del Subsistema Inferior. En primer lugar, reparte la funcionalidad entre el Módulo de Protocolos y el Módulo de Capa Física. A continuación define las interfaces del Módulo de Protocolos. Por último, se encarga de especificar el Plan de Pruebas.
2. Diseño del Módulo de Protocolos: En esta fase se modelan los elementos que forman parte del Módulo de Protocolos. Realiza el diseño de la estructura y, seguidamente, el diseño detallado de cada elemento. La fase termina una vez se somete el diseño a las Pruebas de Nivel definidas en la fase anterior.
3. Integración del Subsistema Inferior: Tiene como propósito la integración, de forma incremental, de los distintos elementos del Subsistema Inferior. Como resultado se obtiene una entidad ejecutable del Subsistema Inferior.

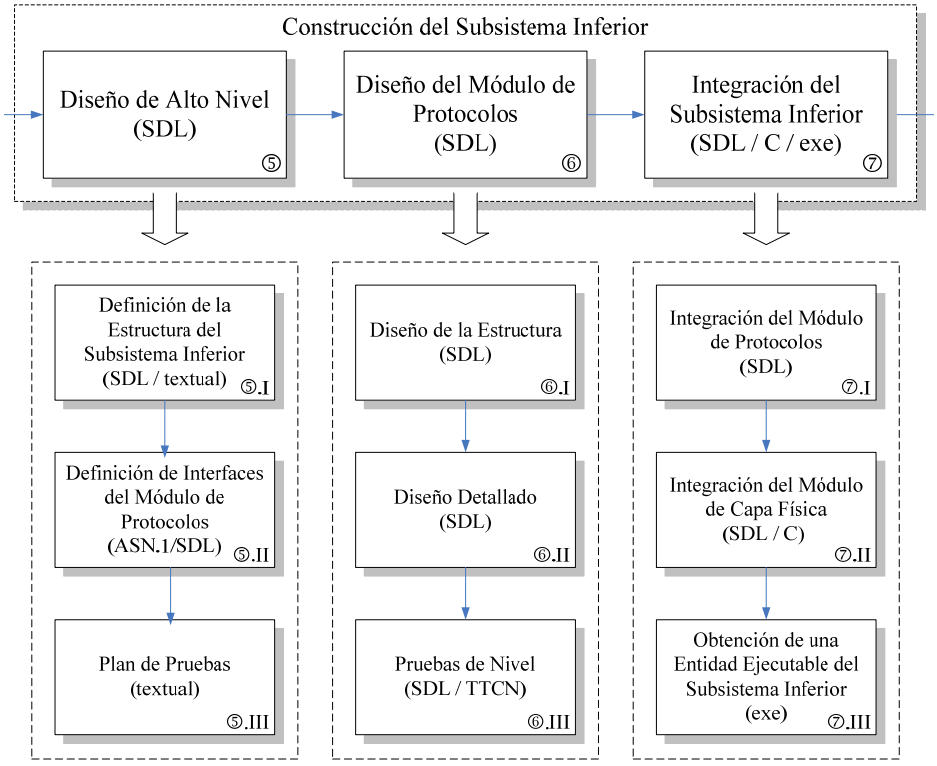


Figura 4.16: Fases para la construcción del Subsistema Inferior.

Existen diversas metodologías que consideran el ciclo de desarrollo completo de sistemas basados en SDL ([OLSE94], [BRÆK93], [REED96], [EKAN95], [BRÆK99], [WITA95]). Para el diseño del Módulo de Protocolos no se ha escogido una metodología específica, sino que su diseño se ha estructurado en una secuencia de actividades, en las que coinciden estas metodologías, e hitos asociados a las mismas. A grandes rasgos, estas actividades son las siguientes:

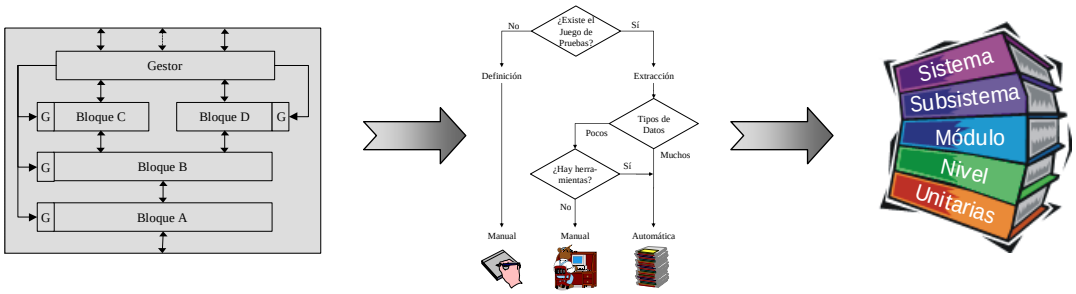
SPICUM servicio de publicaciones

- Análisis de requisitos: Ha sido realizado en las etapas previas.
- Descomposición: División de la funcionalidad del sistema en elementos más pequeños.
- Comportamiento: Modelado del comportamiento de cada elemento.
- Pruebas: Comprobación de la operación del modelo.
- Implementación: Generación de un modelo ejecutable.

En [GEPP01] se define un patrón SDL como "*un artefacto software reutilizable que representa una solución genérica para un problema de diseño recurrente con SDL como el lenguaje de diseño*". En el diseño del Módulo de Protocolos se hace uso de patrones tanto para definir la estructura de alto nivel como para diseñar la estructura interna de un bloque.

4.5.1 Diseño de Alto Nivel

La fase Diseño de Alto Nivel tiene como objetivo principal fijar la estructura del Subsistema Inferior y sus interfaces. Toma como base la arquitectura descrita en el Capítulo 3, Sección 3.1, donde se identifican cada uno de los elementos que componen el Sistema de Pruebas así como las vías de interacción entre estos elementos. Esta fase no tiene como objetivo el modelado del comportamiento de ninguno de los elementos que forman parte del Subsistema Inferior; esta labor es objeto de fases posteriores. En esta etapa se generan los primeros modelos del Módulo de Protocolos, que posteriormente se irán refinando; esto requiere decidir el lenguaje de modelado que se va a emplear.



Definición de la Estructura del Subsistema Inferior

Definición de Interfaces del Módulo de Protocolos

Plan de Pruebas

Figura 4.17: Actividades de la fase Diseño de Alto Nivel.

En esta fase se realizan las siguientes actividades (Figura 4.17):

a) Definición de la Estructura del Subsistema Inferior: En esta actividad, en primer lugar se decide la división funcional entre el Módulo de Protocolos y el Módulo de Capa Física. Una vez hecho, se define la estructura interna del Módulo de Protocolos y se asignan responsabilidades a cada uno de sus componentes.

b) Definición de Interfaces del Módulo de Protocolos: El Módulo de Protocolos posee tanto interfaces externas (Subsistema de Pruebas, Módulo de Capa Física) como internas (entre bloques). En esta actividad se definen las señales que se comunican por estas interfaces y la información que transportan.

SPICUM servicio de publicaciones

c) Plan de Pruebas: Consiste en la especificación de todas las pruebas que se van a realizar a lo largo del diseño del Subsistema Inferior y la posterior integración del Sistema de Pruebas.

4.5.1.1 Definición de la Estructura del Subsistema Inferior

El Subsistema Inferior se divide en Módulo de Protocolos y Módulo de Capa Física. La estructura de estos módulos viene determinada por los Métodos de Pruebas a emplear. Las tareas que hay que realizar en esta subfase, cuya descripción se incluye a continuación, son las siguientes (Figura 4.18):

a) División funcional entre el Módulo de Protocolos y el Módulo de Capa Física;

b) Definición de la estructura del Módulo de Protocolos;

c) Asignación de responsabilidades a los subcomponentes.

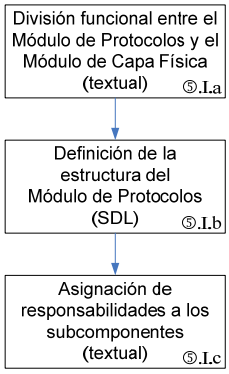


Figura 4.18: Actividades en la subfase Definición de la Estructura del Subsistema Inferior.

4.5.1.1.1 División funcional entre el Módulo de Protocolos y el Módulo de Capa Física

El acceso al medio físico está sometido habitualmente a fuertes requisitos temporales, que hacen de estas funciones elementos críticos pues deben ejecutarse en instantes de tiempo muy concretos y su duración debe ser corta. Por ejemplo, un equipo Bluetooth debe ser capaz de recibir y transmitir datos cada 1,25 ms, dividido en un intervalo de recepción y otro de transmisión de 612,5 μs cada uno (Figura 4.19).

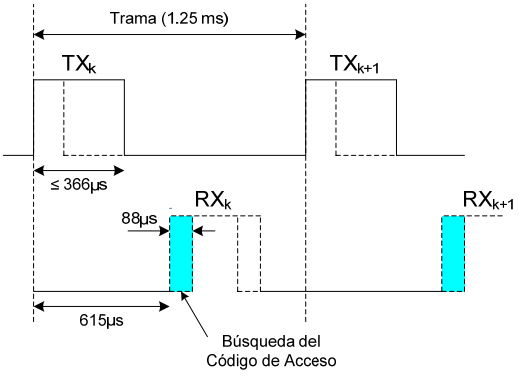


Figura 4.19: Ciclo básico de recepción y transmisión de un equipo Bluetooth [SONN98].

SPICUM servicio de publicaciones

Dado que la comunicación entre el Módulo de Capa Física y el Módulo de Protocolos es en esencia una comunicación con un elemento remoto, donde es complejo conseguir una sincronización estricta, es conveniente incluir en el Módulo de Capa Física todas aquellas funciones que tengan estrictos requisitos temporales, con una alta frecuencia y que requieran intervalos cortos para su ejecución; es decir, aquellas funciones que requieran una fuerte sincronización. En el Módulo de Protocolos se incluye el resto de la funcionalidad del Subsistema Inferior.

Por tanto, el principio que guía esta división funcional es la sincronización temporal. La frontera entre ambos módulos se encontrará en el nivel de acceso al medio, cuya funcionalidad puede incluirse en uno u otro Módulo dependiendo de los requisitos concretos de sincronización o de los elementos comerciales que se utilicen.

4.5.1.1.2 Definición de la estructura del Módulo de Protocolos

El Módulo de Protocolos está constituido por aquellos niveles del Subsistema Inferior necesarios para los Juegos de Pruebas, salvo la funcionalidad incluida en el Módulo de Capa Física. Por este motivo, su estructura será en esencia, la misma que la que venga fijada en la Especificación de Sistema base, la cual normalmente está organizada en niveles.

La definición de la estructura exige decidir el lenguaje en que se va a modelar el Módulo de Protocolos. Siguiendo el principio enunciado de utilizar lenguajes de la familia ITU, pensados para el diseño, especificación e implementación de sistemas de telecomunicación, se ha decidido utilizar el lenguaje SDL. Así, cada uno de los niveles se traduce en un bloque de un sistema SDL[17], como se muestra en la Figura 4.20. El número de canales que posea el diseño SDL dependerá de la interfaz con el exterior del sistema y no aparece reflejado en la figura.

Si se trabaja con diferentes Juegos de Pruebas puede ocurrir que el Módulo de Protocolos sea distinto para cada uno de ellos. Por ejemplo, los Juegos de Pruebas de Bluetooth (Apéndice C, Sección C.4) para los niveles L2CAP [BTEST L2CAP], SPP [BTEST SPP] y SDP [BTEST SDP] requieren tres Módulos de Protocolos distintos[18]. Analizando cada uno de ellos se observa que las diferencias consisten básicamente en la adición o eliminación de alguna de los niveles y tal vez en las posibles interacciones entre ellos.

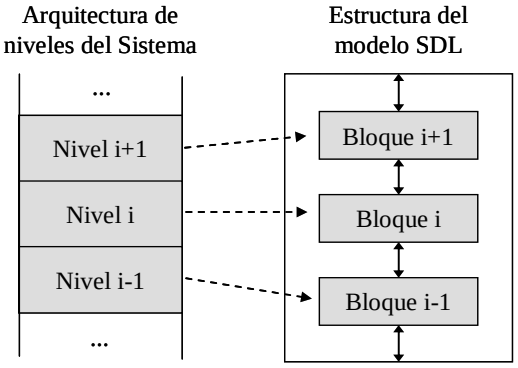


Figura 4.20: Definición de la estructura de bloques del Módulo de Protocolos.

[17] El diseño detallado de cada uno de estos bloques no es labor de esta fase.

[18] Este número aumenta hasta 6 si se incluyen todos los Juegos de Prueba.

SPICUM servicio de publicaciones

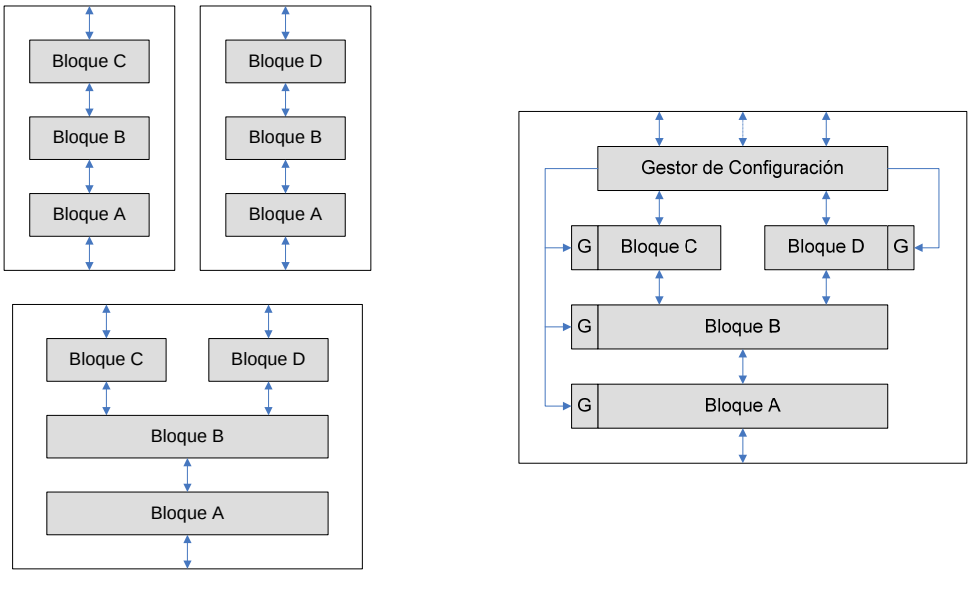
Para realizar la implementación del Módulo de Protocolos se han considerado dos alternativas:

a) Múltiples Módulos de Protocolos: Se implementa un Módulo de Protocolos diferente para cada una Juego de Pruebas.

b) Único Módulo de Protocolos: Se incorpora un mecanismo de selección de la configuración en función del Juego de Pruebas.

Se ha preferido emplear la segunda alternativa, por la mayor eficiencia que ofrece. La implementación de este mecanismo requiere un conjunto adicional de señales para indicar qué configuración debe seleccionarse en el Módulo de Protocolos. La gestión de estas señales se realiza a través de un bloque adicional que se ha denominado `Gestor de Configuración`. El `Gestor de Configuración`, de acuerdo con la configuración solicitada, inicia unos bloques u otros. Los canales empleados en la comunicación entre los bloques y de estos con el exterior son elementos estáticos, siempre definidos, de un diseño SDL, por tanto el `Gestor de Configuración` no necesita realizar acción alguna sobre ellos.

Aunque estáticamente hay que modelar todos los canales de comunicación, dinámicamente algunos de estos canales pueden quedar fuera de uso. El `Gestor de Configuración` es el encargado de que las señales intercambiadas con el entorno del sistema se encaminen por aquellos canales que utilice la configuración seleccionada, lo que conlleva un pequeño incremento en la carga de procesamiento. Se puede asumir que durante la ejecución no habrá señales que utilicen aquellos canales conectados a bloques no inicializados: por un lado, no habrá bloques SDL que generen señales por esos canales y, por otro lado, la configuración elegida establece que no se recibirán señales desde el entorno a través de canales conectados a bloques no inicializados. Si se recibiera alguna señal a través de un canal no activo sería debido a un error de comportamiento del sistema o del entorno.



(a) *(b)*

Figura 4.21: Ejemplo del resultado de emplear diferentes alternativas en la construcción del Subsistema Inferior: (a) Múltiples Módulos de Protocolos; (b) Único Módulo de Protocolos.

SPICUM servicio de publicaciones

La Figura 4.21 muestra un ejemplo de estructura para las dos alternativas planteadas. Supongamos que los Casos de Prueba seleccionados requieren el uso de tres configuraciones diferentes del Módulo de Protocolos, que siempre incluyen a los bloques `A` y `B`, pudiendo también incluir uno o ambos de los bloques `C` y `D`. Con la primera alternativa, los Módulos de Protocolos que deben construirse son los mostrados en la Figura 4.21-a. Si se incorpora un mecanismo interno de configuración (segunda alternativa) se llega a una estructura como la indicada en la Figura 4.21-b. En esta figura se observa la presencia del bloque `Gestor de Configuración`. Cada uno de los bloques restantes incorpora una funcionalidad nueva, adicional, para permitir su configuración; en la figura se indica como una extensión del bloque y se ha rotulado con `G`. Para los bloques `A` y `B` se ha asumido que se requiere algún tipo de indicación sobre el uso de alguna funcionalidad; en caso de no ser así, no sería necesaria esta conexión con el bloque `Gestor de Configuración`. También se incluye un nuevo canal, mostrado en trazo discontinuo, a través del cual se selecciona la configuración.

Tabla 4.6: Ventajas e inconvenientes al utilizar uno o múltiples Módulos de Protocolos.

Alternativa	Ventajas	Inconvenientes
Múltiple	Rápida implementación	Alto coste de mantenimiento
	Adecuado para prototipos	Posible fuente de errores
Único	Inclusión rápida de nuevas configuraciones	Mayor esfuerzo inicial
	Mayor flexibilidad	Sobrecarga adicional al tener que multiplexar las señales

La elección de una u otra de las alternativas consideradas puede depender del objetivo. Si se trata, como en nuestro caso, de la implementación de un Sistema de Pruebas comercial, el balance resulta claramente favorable para la segunda alternativa, pues proporciona una mayor flexibilidad y permite disponer rápidamente de nuevas configuraciones. El mantenimiento es menos costoso, sobre todo si se considera que el coste de un cambio en alguno de los niveles es mayor conforme el proyecto se encuentra más avanzado. Además, permite incluir fácilmente configuraciones de prueba para la verificación del propio desarrollo. Se debe tener también en cuenta que el mantenimiento de múltiples configuraciones puede ser fuente de errores, ya que si se modifica alguno de los bloques comunes, hay que incluir estas modificaciones en todas las configuraciones[19].

4.5.1.1.3 Asignación de responsabilidades a los subcomponentes

En paralelo con la definición de la estructura, hay que determinar la funcionalidad que debe ofrecer cada bloque. La asignación de funcionalidades concretas es una labor que debe realizarse específicamente para cada Sistema de Pruebas, y, en general, estará compuesta por dos partes:

[19] Si existiera una sola copia de los bloques comunes, no existiría la posibilidad de fallos por el uso de distintas versiones en cada configuración. Sin embargo, en un entorno de desarrollo industrial la querencia a utilizar copias de estos bloques puede conllevar que no se actualicen adecuadamente y, por ello, provoque problemas.

SPICUM servicio de publicaciones

a) Funcionalidad básica: Es la funcionalidad del nivel asociado a cada bloque SDL (Figura 4.20). Esta funcionalidad viene recogida en la parte de las Especificaciones de Sistema que define el nivel.

b) Funcionalidad adicional: Hay ocasiones en que las Pruebas solicitan del Sistema de Pruebas un comportamiento especial que no existiría en equipos no destinados a las labores de certificación. Este comportamiento puede influir en la interacción con la IUT o en el procesamiento interno que realice el Módulo de Protocolos. Algunos ejemplos se muestran en la Tabla 4.7. Esta información viene recogida en las Especificaciones de Prueba. También se pueden considerar aquí funcionalidades que el suministrador desea incluir para proporcionar un valor añadido a su producto o para facilitar el desarrollo[20].

Tabla 4.7: Ejemplo de comportamientos especiales de prueba en diferentes sistemas.

Tecnología	Juego de Pruebas	Caso de Prueba	Comportamiento especial del Sistema de Pruebas
UMTS	MAC	`TC_7_1_1_8`	No incluir la cabecera MAC en transmisión
DECT	MAC_PT	`TC_PT_DT_BI_00`	Incluir CRC erróneo en las tramas
Bluetooth	L2CAP	`TP_COS_CED_BV_08`	No contestar a solicitudes de la IUT

4.5.1.1.4 Entradas y Salidas

Las tareas que se realizan en esta subfase son esencialmente de análisis y definición, por lo que las entradas fundamentales son las Especificaciones, tanto de Sistema como de Prueba, y las salidas son documentos, textuales o abstractos, de muy alto nivel.

Entradas	Salidas

• Especificaciones de Sistema
• Especificaciones de Prueba
  o Métodos de Pruebas
• Arquitectura
• Lenguaje SDL [Z.100]
	• Módulo de Protocolos
	  o Descripción funcional
	  ▪ Básico
	  ▪ Adicional
	  o Estructura de bloques

Herramientas	
• Entorno de desarrollo en SDL o Editor • Editor de MSCs	• Reglas de nombrado

Figura 4.22 Entradas y salidas de la subfase Definición de la Estructura del Subsistema Inferior.

4.5.1.2 Definición de Interfaces del Módulo de Protocolos

Para cerrar la definición de la estructura del Módulo de Protocolos, hay que definir las vías de comunicación entre los diferentes módulos, así como el conjunto de señales que

[20] Por ejemplo para facilitar la depuración.

SPICUM servicio de publicaciones

pueden recibirse y enviarse por cada una de ellas y los parámetros que transportan. En concreto, el Módulo de Protocolos posee las siguientes interfaces (Figura 4.23):

a) Interfaz con el Subsistema de Pruebas: Uno o más puntos de acceso que proporcionan los servicios que necesitan las pruebas para su ejecución. Es la interfaz ofrecida en la frontera superior del Subsistema Inferior.

b) Interfaces internas del Módulo de Protocolos: Están constituidas por uno o más canales que permiten comunicar parejas de bloques.

c) Interfaz con el Módulo de Capa Física: Normalmente está formada por un solo punto de acceso. Es una interfaz interna al Subsistema Inferior que se ofrece en la frontera inferior del Módulo de Protocolos.

d) Interfaz adicional de Control: Es una interfaz no estandarizada, para invocar un comportamiento especial en el Módulo de Protocolos o recibir información adicional.

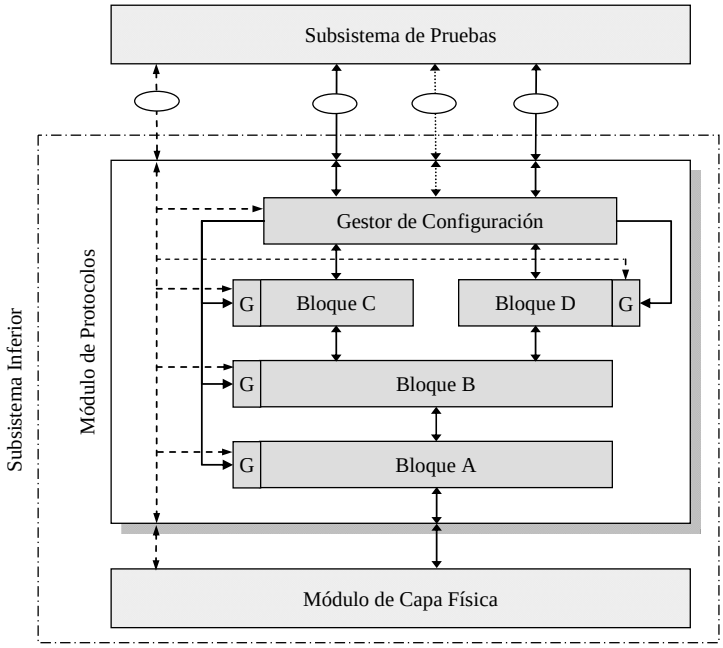


Figura 4.23: Interfaces del Módulo de Protocolos.

4.5.1.2.1 Interfaz con el Subsistema de Pruebas

La interfaz entre el Módulo del Protocolos y el Subsistema de Pruebas (Figura 4.24 y Figura 4.13) está definida por los siguientes elementos:

- Puntos de Control y Observación (PCOs): Estos son los puntos a través de los cuales los Casos de Prueba pueden interactuar con la IUT y estudiar su comportamiento. Aquellos que interaccionen con el Subsistema Inferior se traducen en Puntos de Acceso al Servicio; cada PCO se conecta a un canal SDL.
- Primitivas (ASPs): Son los elementos que permiten transportar información a través de los PCOs. En SDL se convierten a señales[21].

[21] El concepto de primitiva en TTCN es análogo al de señal en SDL, con la salvedad de que todos los campos de una primitiva son opcionales por defecto, mientras que en SDL hay que indicar cuáles son los campos opcionales.

SPICUM servicio de publicaciones

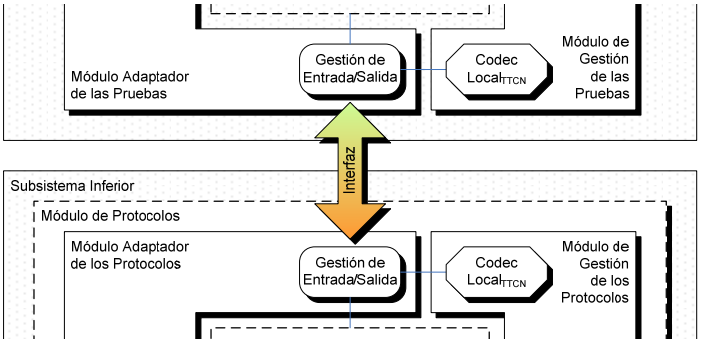


Figura 4.24: Interfaz entre el Subsistema de Pruebas y el Módulo de Protocolos.

- Tipos de datos: Definición de la información transportada por las primitivas. El Juego de Pruebas puede definir tipos de datos que no se usan en la interacción con el exterior de las mismas, por lo que será necesario seleccionar aquellos relevantes para el Subsistema Inferior.

Al ser una interfaz entre dos módulos descritos en TTCN y SDL, respectivamente, la notación más adecuada para definir las primitivas y tipos de datos que se intercambian es ASN.1.

Existen dos alternativas para generar esta interfaz en el Módulo de Protocolos:

a) Definición a partir de las Especificaciones de Sistema. Se trata de definir manualmente las señales de la interfaz a partir de la información contenida en las Especificaciones de Sistema.

Tabla 4.8: Características de las alternativas para generar la interfaz entre el Subsistema de Pruebas y el Módulo de Protocolos.

Vía	Método	Fuente	Complejidad	Conversor	Comentarios
Definición	Manual	Especificaciones de Sistema	Alta	Sí	Puede ser complicado efectuar la correspondencia entre ambos conjuntos de señales, principalmente por la semántica que pueda estar involucrada, ya que se ha modelado por dos vías diferentes.
Extracción	Manual	Juegos de Pruebas	Media	No	Es tedioso y, en caso de cambios, obliga a rehacer el trabajo.
Extracción	Automática	Juegos de Pruebas	Baja	No	Requiere una herramienta adicional que lea los Juegos de Pruebas y extraiga la información correspondiente. Permite una actualización inmediata de posibles cambios. El proceso de extracción es sencillo.

b) Extracción a partir de los Juegos de Pruebas. Esta vía se basa en la idea de que, si ya están definidas las señales, así como sus parámetros, ¿por qué rehacer nuevamente el trabajo? Los diseñadores de los Juegos de Pruebas han definido la interfaz a partir de las Especificaciones de Sistema, complementándola con

SPICUM servicio de publicaciones

señales y parámetros exclusivos de las Pruebas. En este caso, la generación de la interfaz se puede realizar de forma automática.

La alternativa más eficiente es la extracción de la interfaz a partir de los Juegos de Pruebas. Además, si en el modelado de los Juegos de Pruebas se han realizado modificaciones, como por ejemplo variar los parámetros de las señales, añadir nuevas señales o no utilizar alguna de las existentes, estos cambios se muestran automáticamente en la interfaz. Para generar esta interfaz se ha construido la herramienta *GenDef* (Capítulo 5, Sección 5.5.2.1). El procedimiento de generación automática requiere disponer de los Juegos de Pruebas previamente, algo que no siempre ocurre. Su ausencia se puede soslayar[22] como se ha indicado en la Sección 4.4.1.

La Figura 4.24 esquematiza las características esenciales de cada una de las alternativas, y la Figura 4.25 describe el proceso de elección de una alternativa u otra. Como se muestra, en caso de disponer de los Juegos de Pruebas se puede optar por generar la interfaz de forma automática o manual. La segunda opción sólo es práctica en caso de que el conjunto de tipos de datos implicados sea reducido.

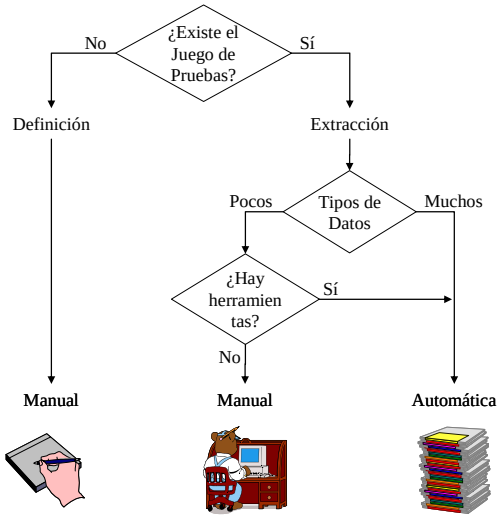


Figura 4.25: Selección del procedimiento para generar la interfaz entre el Subsistema de Pruebas y el Módulo de Protocolos.

4.5.1.2.2 Interfaces internas del Módulo de Protocolos

En esta fase se deben definir, basándose en la división de funcionalidad realizada con anterioridad, los siguientes elementos:

- Canales entre cada par de bloques (Figura 4.23).
- Primitivas que puede transportar cada uno de estos canales: Esta definición debe incluir la semántica asociada a cada una de las primitivas.

[22] De no ser así, habría que optar por la definición manual y decidir en algún momento futuro si adaptar la interfaz a la que se haya definido en los Juegos de Pruebas o bien incluir un módulo conversor. Este camino presenta una mayor complejidad en el desarrollo, además de poder provocar una pérdida de eficiencia del producto, al tener que realizar conversiones adicionales.

SPICUM servicio de publicaciones

- Parámetros que posee cada primitiva: Incluyendo la semántica asociada a cada uno.

Estas interfaces son propias, sin estar obligadas a seguir los dictados de ninguna especificación, siempre y cuando el servicio que se proporcione al Subsistema de Pruebas sea el adecuado. A pesar de ello, la definición de la interfaz con el Subsistema de Pruebas limita la libertad de estas interfaces. Por ejemplo, aunque es posible utilizar como parámetros tipos de datos diferentes de los utilizados en la interfaz anterior, parece sensato adaptarse a estos allí donde sea posible. Además, las primitivas de las Pruebas, cuando solicitan un servicio, transportan un determinado conjunto de información, por lo que las señales que se generen internamente entre cada bloque tendrán que llevar un conjunto de parámetros acorde con la información original.

La Figura 4.26 muestra un esquema simplificado de un modelo SDL donde un proceso `Inicializador` recibe los comandos iniciales por el punto de acceso superior. La Figura 4.26-a muestra la estructura, donde la interfaz de acceso al servicio se divide internamente en 3 canales, cada uno de los cuales enlaza con un proceso diferente. En la Figura 4.26-b se muestra un ejemplo de copia de señales. La señal `Inic`, que contiene la información necesaria para configurar el transmisor y el receptor es recibida por el proceso `Inicializar`, el cual genera las señales `Inic_Rx` e `Inic_Tx` para configurar, respectivamente, al receptor y al transmisor; los parámetros de la señal original son copiados en estas nuevas señales.

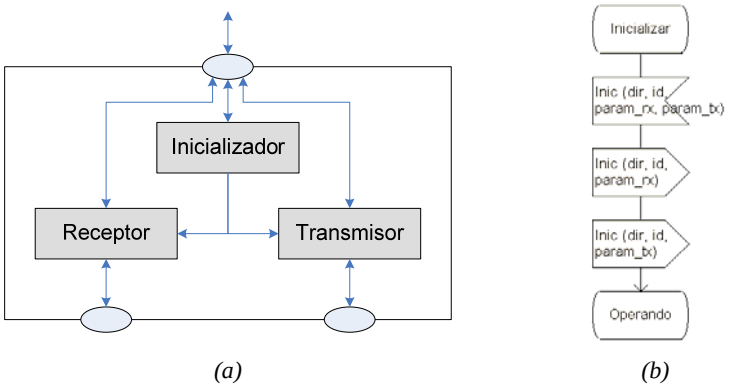


Figura 4.26: (a) Modelo SDL de ejemplo y (b) esquema de copia y reenvío de señales.

4.5.1.2.3 Interfaz con el Módulo de Capa Física

La interfaz entre el Módulo de Protocolos y el Módulo de Capa Física es una interfaz interna al Subsistema Inferior. Para su definición, aunque se trata de una interfaz local, es recomendable seguir las indicaciones incluidas en las Especificaciones de Sistema. Además de la funcionalidad habitual de transferencia de datos y señalización, es probable que esta interfaz incluya primitivas de sincronización[23] no definidas en las Especificaciones de Sistema.

[23] Estas primitivas permiten que la operación del Módulo de Protocolos se sincronice con el Módulo de Capa Física, cuya operación se encuentra restringida por la sincronización de las tramas del medio físico (acceso radio en sistemas de comunicaciones inalámbricas).

SPICUM servicio de publicaciones

4.5.1.2.4 Interfaz adicional de Control

La interfaz adicional de Control se utiliza para invocar modos particulares de operación en el Subsistema Inferior, tanto en el Módulo de Protocolos como en el Módulo de Capa Física. En concreto, esta interfaz se ha pensado para poder inicializar el Subsistema de Pruebas, seleccionar la configuración del Módulo de Protocolos, activar comportamientos especiales requeridos por los Casos de Prueba y para transmitir cualquier otra información que el suministrador considere adecuada[24]. Al igual que la interfaz con el Subsistema de Pruebas, lo más adecuado es definir las primitivas y tipos de datos en ASN.1.

Esta interfaz se modela como un nuevo punto de acceso con objeto de no distorsionar el servicio ofrecido por los puntos de acceso estándares[25]. El canal del Módulo de Protocolos que corresponde a este punto de acceso se ha denominado `INTERNO` (Figura 4.27). Las señales transportadas por el canal `INTERNO` pueden irse añadiendo conforme se vaya determinando su necesidad; en este punto es complicado definir una lista exhaustiva pues se requiere un análisis detallado de todas y cada uno de los Casos de Prueba.

El nuevo punto de acceso conecta el Subsistema de Pruebas con todos los bloques del Subsistema Inferior que puedan necesitarlo. La conexión con el Módulo de Capa Física se realiza siempre atravesando el Módulo de Protocolos, ya que la arquitectura utilizada no proporciona comunicación directa entre el Subsistema de Pruebas y el Módulo de Capa Física.

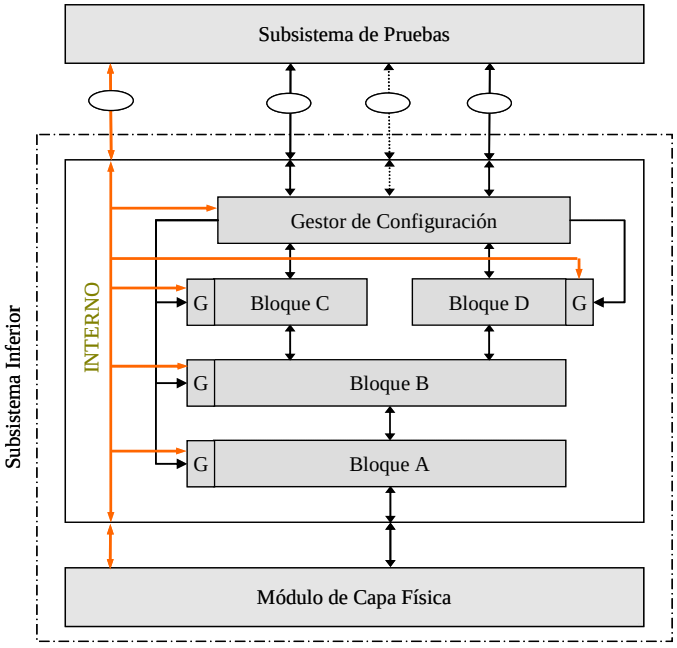


Figura 4.27: Inclusión del canal `INTERNO` en el Subsistema Inferior.

[24] Por ejemplo, la obtención de información para la ejecución de un Caso de Prueba o la realización de verificaciones internas del comportamiento del Subsistema Inferior.

[25] Puede utilizarse más de un punto de acceso si se considera adecuado.

SPICUM servicio de publicaciones

4.5.1.2.5 Entradas y Salidas

Las entradas de esta fase son aquellas necesarias para definir las interfaces del Módulo de Protocolos.

Tabla 4.9: Entradas y salidas de la subfase Definición de Interfaces del Módulo de Protocolos.

Entradas	Salidas

• Especificaciones de Sistema
• Juegos de Pruebas
• Módulo de Protocolos
  o Descripción funcional
  o Estructura de bloques
• Notación ASN.1 ([X.680], [Z.105])
	• Módulo de Protocolos
	  o Interfaces (señales y parámetros)
	  ▪ Subsistema de Pruebas
	  ▪ Internas
	  ▪ Módulo de Capa Física
	  ▪ Adicional de Control

Herramientas	
• Editor de textos	• Generador de Interfaces o Generador de la Definición de la Interfaz • Reglas de nombrado

4.5.1.3 Plan de Pruebas

El Plan de Pruebas es la especificación de las comprobaciones que se van a realizar desde el inicio del desarrollo hasta la obtención del producto. El esfuerzo de las pruebas no debe centrarse únicamente en el resultado final, sino que debe incidir en los múltiples elementos que se vayan obteniendo a lo largo del proceso. De esta forma, cada etapa ofrecerá un alto grado de confianza a las siguientes. Una clasificación generalmente aceptada de las pruebas se presenta en ([OSTR94], [GLAS93]); para las pruebas de las actividades en que se integran distintos componentes se describen varias estrategias en [WHIT94].

Figura 4.28: Categorías de pruebas en el Plan de Pruebas.

El Plan de Pruebas contiene cinco categorías de pruebas (Figura 4.28), con complejidad y ámbito crecientes, con objeto de adaptarse a las diversas fases conforme avanza el

SPICUM servicio de publicaciones

desarrollo. Para cada prueba se debe incluir su objetivo y el resultado esperado. Las categorías de pruebas son las siguientes:

a) Pruebas Unitarias: Son pruebas realizadas sobre componentes relativamente pequeños del sistema completo, como ramas del comportamiento, bucles, procedimientos e, incluso, procesos, cuya unión permite obtener la funcionalidad esperada de un nivel[26]. Este tipo de pruebas se realizan en un simulador.

b) Pruebas de Nivel: Se encargan de verificar que la funcionalidad proporcionada por cada uno de los bloques del Módulo de Protocolos es la esperada; su complejidad depende de la complejidad del propio bloque. Este tipo de prueba permite comprobar las interacciones entre aquellos componentes que han sido verificados en la categoría anterior. Son un requisito necesario antes de poder continuar el desarrollo con el fin de evitar posibles fuentes de error al integrar el sistema. Al igual que las pruebas unitarias, se realizan en un simulador. En la Sección 4.5.2.3 se comentan estas pruebas en mayor detalle.

c) Pruebas de Módulo: Este conjunto de pruebas está pensado para verificar la integración del Módulo de Protocolos. Se comprueba así que la interacción entre cada uno de los bloques que constituyen el Módulo de Protocolos se realiza correctamente.

d) Pruebas de Subsistema: El objetivo de esta categoría de pruebas es la verificación del Subsistema Inferior una vez se han integrado el Módulo de Protocolos y el Módulo de Capa Física. Este tipo de pruebas permiten comprobar aspectos de sincronización y de prestaciones y son pruebas de tiempo real.

e) Pruebas de Sistema: Se trata de las pruebas finales, en las que se comprueba el Sistema de Pruebas como un todo. Este grupo de pruebas tiene dos vertientes. En primer lugar, comprobar el correcto funcionamiento de la integración de los diferentes Subsistemas que forman el Sistema de Pruebas. En segundo lugar, validar este funcionamiento mediante el uso de equipos comerciales que sean enfrentados al Sistema de Pruebas con objeto de determinar si los Casos de Prueba incluidos se ejecutan adecuadamente[27]. Formalmente, la definición del Plan de Pruebas debe hacerse en la fase Definición del Sistema de Pruebas, pero se ha incluido aquí por englobar la especificación de todas las pruebas a realizar sobre los componentes del Sistema de Pruebas (y él mismo) en una única fase.

Para realizar las pruebas es conveniente hacer uso de las herramientas de simulación que proporcione el entorno de desarrollo. El objetivo es no tener que generar aplicaciones independientes de este entorno hasta la fase de diseño más avanzada posible. En concreto, hasta la realización de las pruebas de sistema. Para la especificación de las pruebas y el análisis de los resultados son muy útiles los diagramas de secuencia de mensajes (MSCs).

[26] Las pruebas que se realicen sobre los procesos pueden verse como pruebas de una calidad superior (en cuanto a complejidad y ámbito) a aquellas pruebas realizadas sobre procedimientos u otras partes del comportamiento. A pesar de ello, no se han separado en una nueva categoría.

[27] Se debe comprobar la ejecución de todos los Casos de Prueba incluidos en el Sistema de Pruebas.

SPICUM servicio de publicaciones

4.5.1.3.1 Entradas y Salidas

La entrada de esta fase es básicamente, una descripción funcional, tanto a nivel de sistema como a nivel de componentes más pequeños. Junto con las interfaces de estos elementos, se puede definir un Plan de Pruebas adecuado.

Tabla 4.10: Entradas y salidas de la subfase Plan de Pruebas.

Entradas	Salidas

• Especificaciones de Sistema
• Especificaciones de Prueba
• Arquitectura
• Módulo de Protocolos
  o Descripción funcional
  o Estructura de bloques
  o Interfaces
	• Plan de Pruebas

Herramientas	
• Editor de textos • Editor de MSCs	

4.5.1.4 Entradas y Salidas

Como se ha visto, la fase Diseño de Alto Nivel es esencialmente una fase de análisis, donde, como resultado, se definen la funcionalidad del Módulo de Protocolos, su estructura y las interfaces que utiliza.

Tabla 4.11: Entradas y salidas de la fase Diseño de Alto Nivel.

Entradas	Salidas

• Especificaciones de Sistema
• Especificaciones de Prueba
  o Métodos de Pruebas
• Juegos de Pruebas
• Arquitectura
• Lenguaje SDL [Z.100]
• Notación ASN.1 ([X.680], [Z.105])
• Lenguaje MSC [Z.120]
	• Módulo de Protocolos
	  o Descripción funcional
	  ▪ Básico
	  ▪ Adicional
	  o Estructura de bloques
	  o Interfaces (señales y parámetros)
	  ▪ Subsistema de Pruebas
	  ▪ Internas
	  ▪ Módulo de Capa Física
	  ▪ Adicional de Control
	• Plan de Pruebas

Herramientas	

• Entorno de desarrollo en SDL
  o Editor
• Editor de textos
• Editor de MSCs
	• Generador de Interfaces
	  o Generador de la Definición de la Interfaz
	• Reglas de nombrado

SPICUM servicio de publicaciones

4.5.2 Diseño del Módulo de Protocolos

La fase Diseño del Módulo de Protocolos tiene como objetivo modelar los elementos que forman parte de dicho módulo, tomando como punto de partida las salidas de la fase anterior (Diseño de Alto Nivel). El lenguaje utilizado para este modelado es SDL. Se trata de una fase costosa en tiempo y esfuerzo, gran parte del cual se dedica a la verificación de los modelos que se realicen. Durante el trabajo de esta fase, se pueden detectar errores en las salidas de la fase anterior, lo que obliga a realizar modificaciones dentro de una visión iterativa del proceso de diseño.

Figura 4.29: Actividades de la fase Diseño del Módulo de Protocolos.

Esta fase se ha dividido en tres actividades (Figura 4.18):

a) Diseño de la Estructura: Se definen aquí los procesos y procedimientos que conforman la estructura interna del Módulo de Protocolos, así como su funcionalidad e interfaces.

b) Diseño Detallado: Actividad se modela el comportamiento de las entidades identificadas durante el Diseño de la Estructura.

c) Pruebas de Nivel: Se verifica el funcionamiento de los modelos desarrollados en las actividades anteriores.

4.5.2.1 Diseño de la Estructura

En esta actividad se define la estructura interna del Módulo de Protocolos. A partir del análisis de la funcionalidad asignada a cada bloque se definen los correspondientes procesos y procedimientos con objeto de realizar una implementación tan modular y estructurada como sea posible[28]. Para cada uno de ellos debe indicarse la funcionalidad de que serán responsables y su interfaz. En el caso de los procesos hay que definir qué señales puede recibir y cuáles enviar; para los procedimientos hay que indicar los parámetros de entrada y el resultado esperado. Como se observa, se trata de una fase donde el énfasis se pone en el análisis y no en la codificación en sí.

Si un bloque está compuesto por un proceso cuyas instancias se crean dinámicamente en función del número de comunicaciones extremo-extremo que se encuentran abiertas, un diseño muy flexible es el que se muestra en la Figura 4.30. El proceso `Gestor` recibe las peticiones de establecimiento y, si corresponde, crea una instancia del proceso controlador del protocolo, que ejecuta la máquina de estados del mismo.

[28] En general, la mayoría de los procesos que se definan corresponderán a un grupo funcional claramente identificado en las Especificaciones de Sistema.

SPICUM servicio de publicaciones

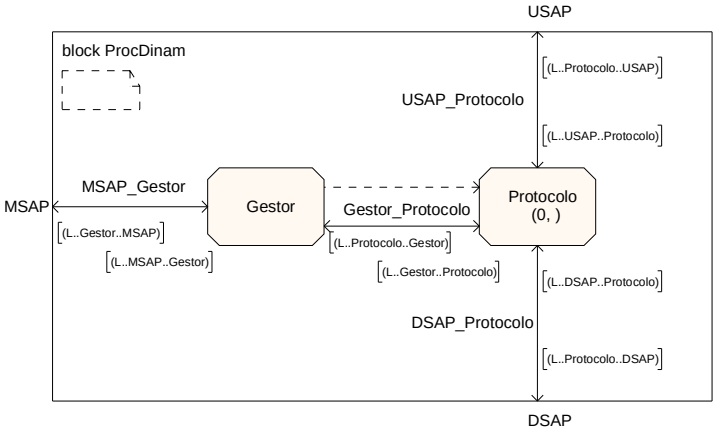


Figura 4.30: Estructura genérica para bloques con procesos de creación dinámica.

4.5.2.1.1 Orientación a Objetos

SDL es un lenguaje que inherentemente integra los conceptos de orientación a objetos desde la versión SDL'92 [OLSE94], pero ¿hasta qué punto merece la pena utilizar estas características en el diseño? La experiencia adquirida a lo largo de este trabajo apunta a que la orientación a objetos en el diseño de Sistemas de Pruebas es útil a la hora de definir instancias del mismo objeto que deban existir concurrentemente, de forma que cada una de dichas instancias controle una entidad. Sin embargo, conceptos como la redefinición o la herencia pueden llevar a un diseño complejo si son utilizados a nivel de bloque o sistema (Figura 4.31). Esto es debido a que en el diseño de Sistemas de Pruebas las oportunidades de reutilización del código, principal ventaja de estos conceptos, son escasas, incluso si se diseñan tanto la estación base como el móvil. Por dicho motivos, no se han encontrado razones para recomendar el uso de estos conceptos.

4.5.2.1.2 Procesos o Servicios

Procesos y servicios son construcciones que permiten modelar la estructura interna de un bloque. Mientras que los procesos son concurrentes entre sí, si un bloque se subdivide en varios servicios, sólo uno de ellos se puede estar ejecutando en cada momento. A partir de la versión SDL-2000 [REED01] se ha considerado que la distinción semántica entre ambas construcciones no es de interés, por lo que el concepto de servicio se ha eliminado.

Sin embargo, la introducción de una nueva versión es un proceso que dura varios años, hasta que es incorporada en los entornos de desarrollo, lo que ha hecho que la versión SDL'92 se haya estado utilizando hasta hace poco tiempo. Por ello, se consideró interesante estudiar el uso de servicios en lugar de procesos para modelar el comportamiento de un bloque [COLA02]; no sólo las implicaciones que tiene su semántica, sino también cómo eran manejados por los entornos de desarrollo. Este análisis se describe en el Capítulo 10, Sección 10.2.2.1. Como resultado, se consideró que, para nuestras necesidades, era más adecuado el uso de procesos. Las conclusiones que se han obtenido se muestran en la Tabla 4.12.

SPICUM servicio de publicaciones

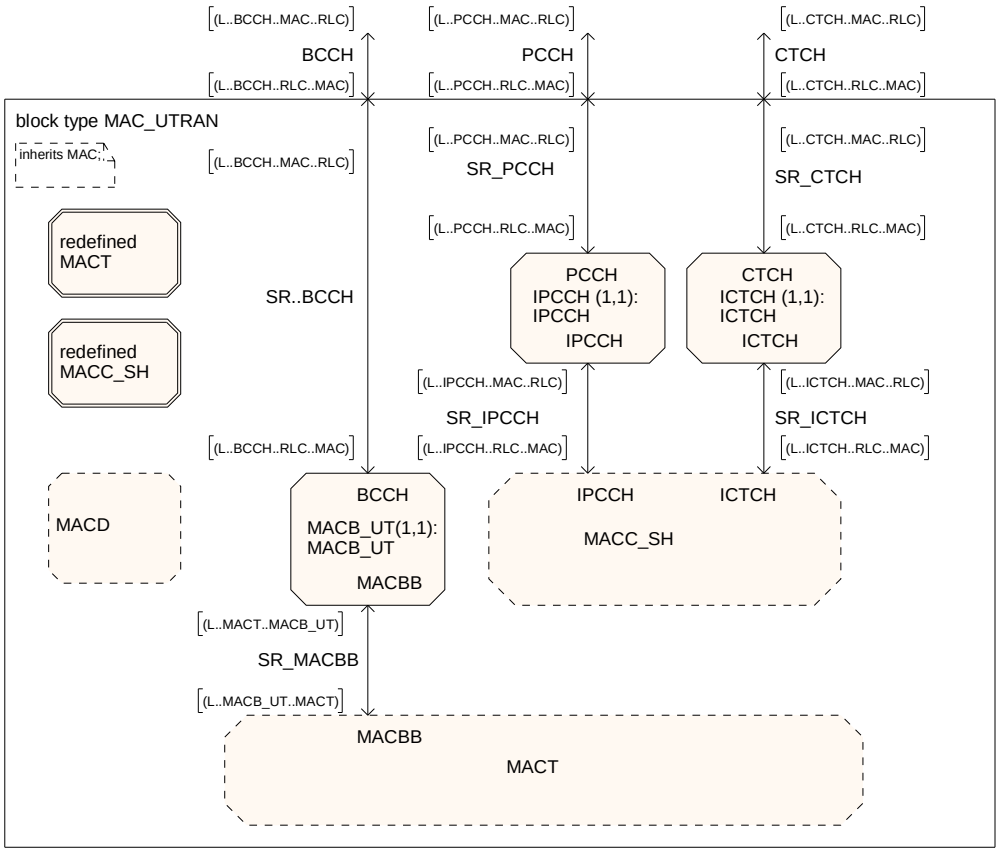


Figura 4.31: Ejemplo de uso del mecanismo de redefinición en el Nivel MAC de UMTS.

Tabla 4.12: Ventajas e inconvenientes del uso de servicios y procesos en SDL.

Servicios	**Procesos**
Ventajas	
Las variables se pueden compartir entre todos los servicios	Se evita la réplica de todo tipo de estructuras
El código generado es más sencillo	Se puede hacer uso del concepto de 'estado' de SDL
	La gestión de instancias la realiza el código generado por las herramientas
Inconvenientes	
Impide la utilización de los conceptos de orientación a objetos, impidiendo al mismo tiempo el uso del concepto de 'estado'; se necesita mantener este 'estado' en variables	La compartición de variables requiere la definición de señales específicas que las transporten
La instanciación de objetos se debe sustituir por la gestión de réplicas del conjunto de variables requerido por uno cualquiera de dichos objetos	El procedimiento de exportación de variables es lento si tienen un tamaño grande

SPICUM servicio de publicaciones

4.5.2.1.3 Entradas y Salidas

La salida de esta subfase es una definición de la estructura del Módulo de Protocolos que organice la funcionalidad en componentes dentro de la estructura de bloques definida en la fase anterior.

Tabla 4.13: Entradas y salidas de la subfase Diseño de la Estructura.

Entradas	Salidas

• Módulo de Protocolos
  o Descripción funcional
  o Estructura de bloques
  o Interfaces
• Lenguaje SDL [Z.100]
• Notación ASN.1 ([X.680], [Z.105])
• Lenguaje MSC [Z.120]
	• Módulo de Protocolos
	  o Modelo con procesos y procedimientos

Herramientas	
• Entorno de desarrollo en SDL o Editor • Editor de MSCs	• Reglas de nombrado

4.5.2.2 Diseño detallado

En esta tarea se modela el comportamiento de las entidades (procesos y procedimientos) identificadas previamente. El resultado final depende en gran medida de las preferencias y experiencia del diseñador, aunque su libertad de acción está restringida por las fronteras delimitadas para cada entidad. Algunos principios de diseño que pueden resultar útiles son mantener al mínimo el número de estados de cada proceso y evitar el uso de estados dentro de los procedimientos.

El diseño de un nivel implica la construcción de un codificador para la comunicación extremo-extremo (Codec $E2E_{SDL}$), al mismo nivel, entre el Sistema de Pruebas y la Implementación Bajo Prueba. Se pueden desarrollar herramientas para generar automáticamente este codificador. Un ejemplo es la herramienta *GenCodecAir* (Capítulo 5, Sección 5.5.3).

Al avanzar en el desarrollo se deben ir realizando las pruebas unitarias definidas para cada parte del comportamiento. Para ello, se hace uso de las utilidades de simulación que proporcionan las herramientas. La complejidad de estas pruebas es baja, por lo que configurar el escenario de una cualquiera de ellas no debe presentar excesivas complicaciones. A partir del resultado de estas pruebas se genera el correspondiente Informe de Pruebas Unitarias.

4.5.2.2.1 Entradas y Salidas

El resultado de esta tarea son modelos en SDL de cada nivel que compone el Módulo de Protocolos, cuyo comportamiento ha sido comprobado por partes.

SPICUM servicio de publicaciones

Tabla 4.14: Entradas y salidas de la subfase Diseño Detallado.

Entradas	Salidas

• Especificaciones de Sistema
• Módulo de Protocolos
  o Descripción funcional
  ▪ Básico
  ▪ Adicional
  o Modelo con procesos y procedimientos
• Plan de Pruebas
  o Unitarias
• Lenguaje SDL [Z.100]
• Notación ASN.1 ([X.680], [Z.105])
• Lenguaje MSC [Z.120]
	• Módulo de Protocolos
	  o Modelo SDL de cada nivel
	• Informe de Pruebas
	  o Unitarias

Herramientas	

• Entorno de desarrollo en SDL
  o Editor
  o Analizador sintáctico y semántico
  o Generador de código
  o Simulador
• Entorno de desarrollo en C
  o Editor
  o Compilador
  o Depurador
• Editor de MSCs
	• Generador de Codificador/Descodificador para la Interfaz Aire
	• Reglas de nombrado

4.5.2.3 Pruebas de Nivel

El objetivo de las Pruebas de Nivel es verificar el correcto funcionamiento de los modelos desarrollados en la subfase Diseño Detallado. Como resultado principal se obtiene un nivel sin errores.

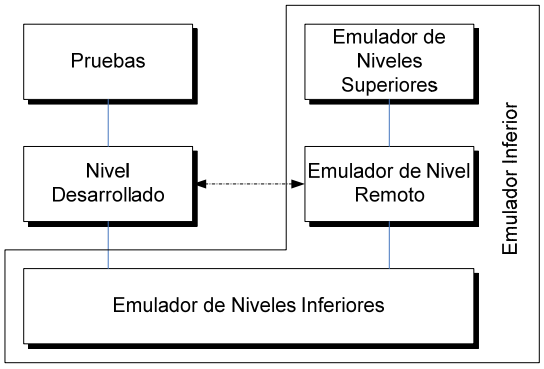


Figura 4.32: Arquitectura de pruebas para comprobar el modelo de un nivel.

Para realizar estas pruebas es necesario comprobar que la interacción del nivel desarrollado, que actúa como nivel local, con un nivel remoto se realiza correctamente.

SPICUM servicio de publicaciones

Para ello, el Método de Pruebas contiene los bloques mostrados en la Figura 4.32: `Pruebas`, `Nivel Desarrollado` y `Emulador Inferior`. Salvo el nivel desarrollado, el resto de bloques pueden ser emuladores[29] de la funcionalidad correspondiente. La funcionalidad de cada bloque es la siguiente:

- `Pruebas`: permite seleccionar la prueba a realizar, genera los estímulos necesarios para su ejecución, y recibe y comprueba las respuestas.
- `Emulador Inferior`: engloba el extremo remoto de la comunicación y los correspondientes niveles inferiores. Está formado, a su vez, por los tres siguientes bloques.
- `Emulador de Nivel Remoto`: actúa como la entidad de igual nivel con la que se comunica el nivel desarrollado. Si el nivel desarrollado es simétrico el nivel remoto será una copia suya; si no, hay que construir también el correspondiente emulador, aunque en este caso su complejidad será similar a la del modelo del nivel desarrollado.
- `Emulador de Niveles Superiores`: actúa de terminador y recoge las indicaciones emitidas por el nivel remoto hacia los niveles superiores, generando respuestas adecuadas para la realización de cada prueba. Se puede modelar una rama específica para cada prueba y activarla al inicio de la ejecución de la misma.
- `Emulador de Niveles Inferiores`: se limita a conectar el nivel desarrollado con el nivel remoto. Puede ser tan simple como el modelado de un cable, realizando tan sólo la función de transportar las primitivas recibidas por una de sus fronteras hasta la frontera opuesta, copiando los parámetros recibidos o transformándolos adecuadamente. Un ejemplo para el nivel físico de UMTS se muestra en la Figura 4.33.

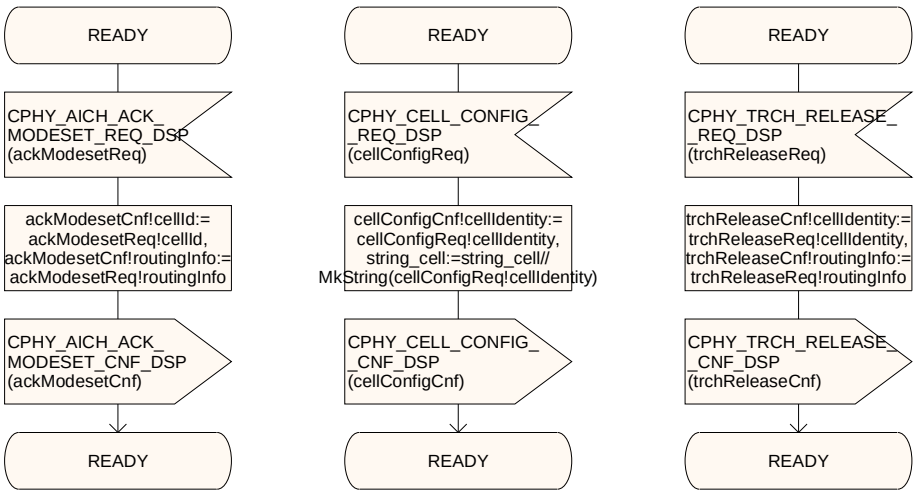


Figura 4.33: Ejemplo de emulador de nivel físico para UMTS.

[29] Un emulador ofrece al exterior la apariencia de implementar toda la funcionalidad, pero sólo necesita responder adecuadamente a los estímulos programados y su comportamiento interno puede simplificarse.

SPICUM servicio de publicaciones

Para construir el bloque `Pruebas` existen dos alternativas. Se puede diseñar un conjunto de pruebas propias (Figura 4.34), con su propósito y su secuencia de eventos, o bien se puede utilizar el Juego de Pruebas correspondiente al nivel desarrollado (Figura 4.35-a). Si el nivel desarrollado se sitúa en el mismo extremo que las pruebas (Figura 4.34 y Figura 4.35-a) en realidad no se está comprobando el nivel desarrollado en sí, sino su funcionamiento como parte del Sistema de Pruebas, evitando un esfuerzo adicional en la construcción del Módulo de Protocolos en fases posteriores. Si se quiere someter el nivel desarrollado al Juego de Pruebas correspondiente hay que emplear el método de pruebas indicado en la Figura 4.35-b.

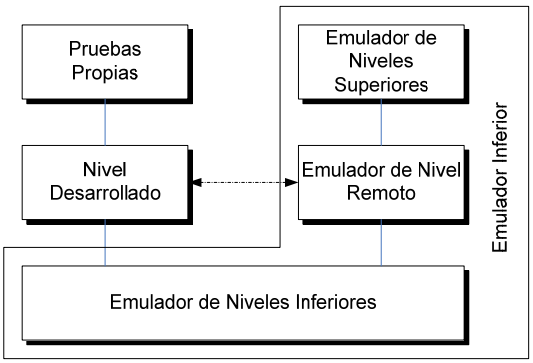


Figura 4.34: Método de prueba con pruebas propias.

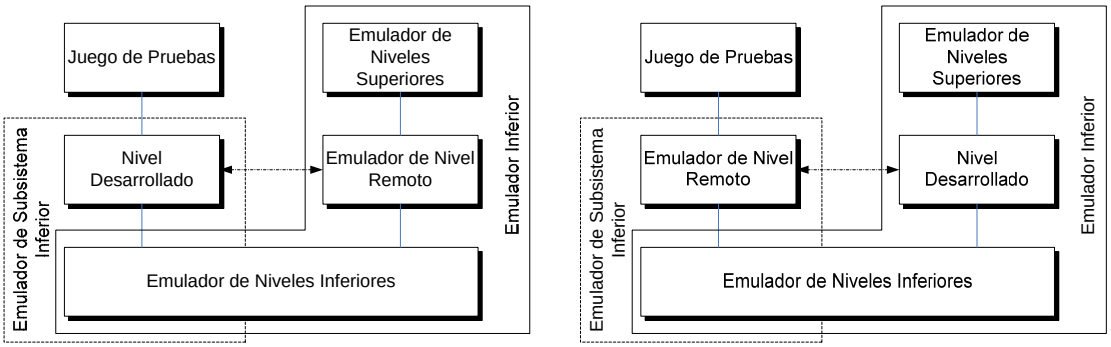


Figura 4.35: Métodos de prueba alternativos al usar Juegos de Pruebas: (a) misma configuración que con pruebas propias; (b) configuración para someter el nivel desarrollado del Juego de Pruebas.

Dentro de lo posible se deben realizar estas pruebas en un simulador, para facilitar la depuración[30]. Sin embargo, esto puede depender de las facilidades que ofrezcan las herramientas para comunicar el simulador del Juego de Pruebas (TTCN) con un simulador del resto de bloques (SDL). Con objeto de permitir repeticiones o posteriores ejecuciones de estas pruebas, éstas deben quedar modeladas como archivos de comandos interpretables por el simulador o en un módulo SDL. Un ejemplo de los primeros se muestra en la Figura 4.36. En el segundo caso, se trata de modelar un

[30] La otra alternativa es generar un ejecutable del modelo y realizar las pruebas fuera del entorno de desarrollo. De esta forma se pierden las posibilidades que ofrece este entorno para el análisis de los resultados, que queda a menudo restringido al análisis de trazas incluidas con este objeto.

SPICUM servicio de publicaciones

bloque SDL donde, según la señal recibida del operador, se elija una u otra secuencia de interacciones, es decir, una u otra prueba.

La realización de esta tarea se documenta en un Informe de Pruebas de Nivel donde quede constancia de qué pruebas se han realizado, cuál ha sido el objetivo de cada una de ellas, cuál ha sido el resultado, los cambios necesarios en caso de haber obtenido un resultado negativo, el guión de la prueba (secuencia de eventos) y la traza de la ejecución.

```
# Configurar TBF dwn
Output-To CONFIG_REQ(0) <<Block UTRAN/Block GESTOR_UTRAN>> GESTOR;
Output-To DATA_REQ (1, 0, 400) <<Block UTRAN/Block GESTOR>> GESTOR;
Output-To DATA_REQ (2, 4, 1024) <<Block UE/Block GESTOR>> GESTOR;
Output-To PHY <<Block UTRAN/Block GESTOR_UTRAN>> GESTOR;
Output-To PHY <<Block UE/Block GESTOR_UE>> GESTOR;
Output-To G_CRLC_DL_TBF_CONFIG_REQ ((. 1,'00010'B ,'0'B , ,'0'B , (. ,
      , , , , , , , , , , , , , , .),(. '00001'B,'000011'B,'00001'B .)
      .)) <<Block RLCMAC>> CTRL
Go
```

Figura 4.36: Ejemplo de archivo de comandos.

4.5.2.3.1 Entradas y Salidas

La subfase Pruebas de Nivel tiene como responsabilidad comprobar la corrección de los modelos desarrollados y generar un Informe de Pruebas de Nivel.

SPICUM servicio de publicaciones

Tabla 4.15: Entradas y salidas de la subfase Pruebas de Nivel.

Entradas	Salidas

• Especificaciones de Sistema
• Módulo de Protocolos
  o Descripción funcional
  o Modelo SDL de cada nivel
  o Interfaces
• Plan de Pruebas
  o Nivel
• (opc) Juegos de Prueba
	• Módulo de Protocolos
	  o Modelo SDL comprobado de cada nivel
	• Informes de Pruebas
	  o Nivel
	• Emuladores
	  o Nivel Remoto
	  ▪ Especificación
	  ▪ Modelo SDL
	  o Niveles Superiores
	  ▪ Especificación
	  ▪ Modelo SDL
	  o Niveles Inferiores
	  ▪ Especificación
	  ▪ Modelo SDL
	• Pruebas
	  o (opc) Pruebas Propias
	• Simuladores
	  o Pruebas
	  ▪ (opc) Pruebas Propias
	  ▪ (opc) Juegos de Pruebas
	  o Nivel Desarrollado
	  o Emulador Inferior

Herramientas	

• Entorno de desarrollo en SDL
  o Simulador
• (opc) Entorno de desarrollo en TTCN
  o Generador de código
  o Simulador
• Entorno de desarrollo en C
  o Compilador
  o Depurador
• Editor de MSCs	

4.5.2.4 Entradas y Salidas

La fase Diseño del Módulo de Protocolos es una fase de desarrollo donde, como resultado, se obtiene un modelo SDL probado para cada nivel de interés.

SPICUM servicio de publicaciones

Tabla 4.16: Entradas y salidas de la fase Diseño de Alto Nivel.

Entradas	Salidas

• Especificaciones de Sistema
• Módulo de Protocolos
  o Descripción funcional
  ▪ Básico
  ▪ Adicional
  o Estructura de bloques
  o Interfaces
• Plan de Pruebas
  o Unitarias
  o Nivel
• (opc) Juegos de Prueba
• Lenguaje SDL [Z.100]
• Notación ASN.1 ([X.680], [Z.105])
• Lenguaje MSC [Z.120]
	• Módulo de Protocolos
	  o Modelo SDL comprobado de cada nivel
	• Informes de Pruebas
	  o Unitarias
	  o Nivel
	• Emuladores
	  o Nivel Remoto
	  ▪ Especificación
	  ▪ Modelo SDL
	  o Niveles Superiores
	  ▪ Especificación
	  ▪ Modelo SDL
	  o Niveles Inferiores
	  ▪ Especificación
	  ▪ Modelo SDL
	• Pruebas
	  o (opc) Pruebas Propias
	• Simuladores
	  o Pruebas
	  ▪ (opc) Pruebas Propias
	  ▪ (opc) Juegos de Pruebas
	  o Nivel Desarrollado
	  o Emulador Inferior

Herramientas	

• Entorno de desarrollo en SDL
  o Editor
  o Analizador sintáctico y semántico
  o Generador de código
  o Simulador
• (opc) Entorno de desarrollo en TTCN
  o Generador de código
  o Simulador
• Entorno de desarrollo en C
  o Editor
  o Compilador
  o Depurador
• Editor de MSCs
	• Generador de Codificador/Descodificador para la Interfaz Aire
	• Reglas de nombrado

SPICUM servicio de publicaciones

4.5.3 Integración del Subsistema Inferior

El objetivo de esta fase es integrar los distintos elementos que componen el Subsistema Inferior y generar una entidad ejecutable del mismo. Por este motivo, se ha dividido esta fase en tres pasos (Figura 4.37):

a) Integración del Módulo de Protocolos: La primera etapa de la integración del Subsistema Inferior es integrar los distintos niveles que componen el Módulo de Protocolos y verificar su funcionamiento conjunto.

b) Integración del Módulo de Capa Física: El Módulo de Capa Física se integra en dos pasos. En primer lugar, se integra con el nivel más bajo del Módulo de Protocolos; a continuación, se integra con el Módulo de Protocolos resultante de la actividad anterior.

c) Obtención de una Entidad Ejecutable del Subsistema Inferior: El ejecutable del Subsistema Inferior se obtiene enlazando los distintos elementos que lo constituyen (Figura 4.40). Los codificadores locales son específicos de cada Sistema de Pruebas (Figura 4.4); su construcción se realiza en esta etapa.

Figura 4.37: Actividades de la fase Integración del Subsistema Inferior.

4.5.3.1 Integración del Módulo de Protocolos

Esta actividad está pensada para comprobar la integración de los modelos SDL que componen el Módulo de Protocolos haciendo uso del simulador del entorno de desarrollo. Si no fuera posible utilizar un simulador, la integración debería probarse una vez obtenida la entidad ejecutable del Subsistema Inferior.

Para integrar el Módulo de Protocolos se sugiere emplear una integración incremental tipo *bottom-up* en el caso de que esté compuesto por más de dos niveles. Esto permite ejecutar, sobre el sistema resultante tras cada paso de la integración, las Pruebas de Nivel correspondientes al nivel superior de la integración. La Figura 4.38 muestra un ejemplo de cómo se realizaría esta integración. Con este tipo de integración, la incertidumbre sobre el correcto funcionamiento se ve limitada a una frontera entre dos niveles y no a posibles interacciones entre varios de ellos. Finalizada la integración, se ejecutan las Pruebas de Módulo y se genera el correspondiente Informe de Pruebas de Módulo. Como Pruebas de Módulo se pueden emplear pruebas propias o los Juegos de Pruebas correspondientes a los niveles integrados.

SPICUM servicio de publicaciones

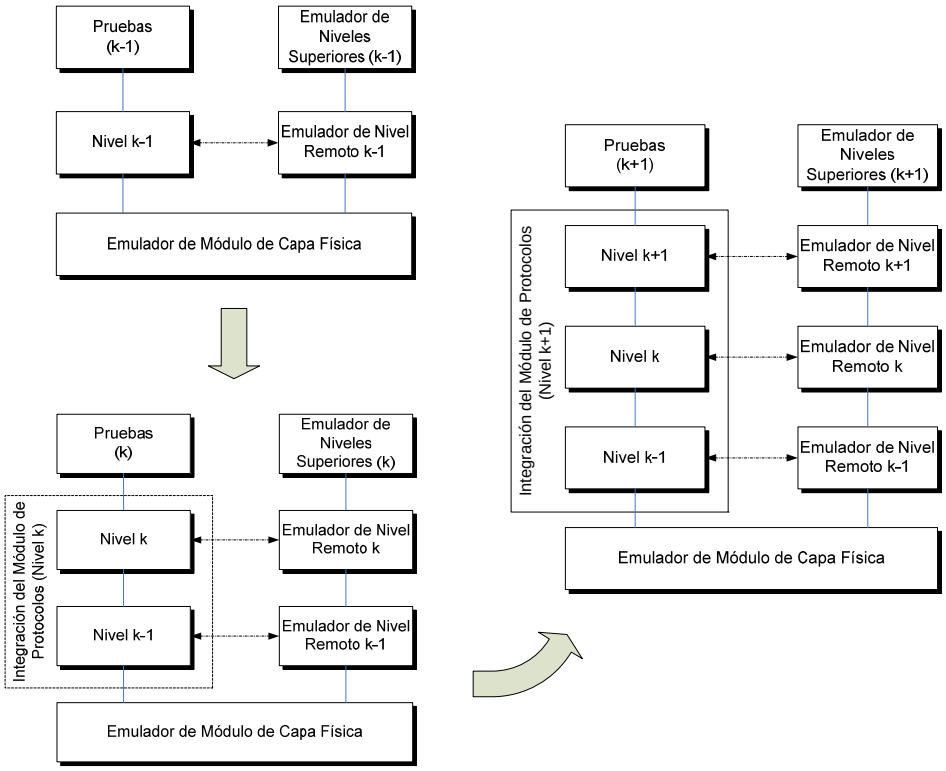


Figura 4.38: Fases de la integración incremental del Módulo de Protocolos.

4.5.3.1.1 Entradas y Salidas

La subfase Integración del Módulo de Protocolos está pensada para comprobar que la integración de los distintos elementos que constituyen el Subsistema Inferior funcione correctamente.

SPICUM servicio de publicaciones

Tabla 4.17: Entradas y salidas de la subfase Integración del Módulo de Protocolos.

Entradas	Salidas

• Especificaciones de Sistema
• Arquitectura
• Módulo de Protocolos
  o Estructura de bloques
  o Modelos SDL de niveles
  o Interfaces
  ▪ Módulo de Capa Física
  ▪ Adicional de Control
  ▪ (opc) Subsistema de Pruebas
• Emuladores
• Plan de Pruebas
  o Módulo
• Pruebas
  o (opc) Pruebas Propias
  o (opc) Juegos de Pruebas
	• Módulo de Protocolos
	  o Integrado
	• (opc) Emulador de IUT
	• Informes de Pruebas
	  o Módulo

Herramientas	

• Entorno de desarrollo en SDL
  o Editor
  o Analizador sintáctico y semántico
  o Generador de código
  o Simulador
• (opc) Entorno de desarrollo en TTCN
  o Simulador
• Entorno de desarrollo en C
  o Editor
  o Compilador
  o Depurador
• Editor de MSCs	

4.5.3.2 Integración del Módulo de Capa Física

Es conveniente realizar la integración del Módulo de Capa Física en dos pasos (Figura 4.39), para tener un mayor control sobre los errores que puedan surgir. En primer lugar, se integra el Módulo de Capa Física con el nivel más bajo del Módulo de Protocolos. Sobre esta integración se ejecutan las pruebas del nivel integrado. A continuación, se integra el Módulo de Capa Física con el Módulo de Protocolos.

Al integrar el Módulo de Capa Física se produce por primera vez una separación real (física), y no sólo lógica, entre los elementos que forman parte del Sistema de Pruebas y los elementos que se están empleando para probar los modelos desarrollados. Esta separación hace necesario utilizar un Motor SDL de tiempo real.

La integración requiere construir un bloque de Entrada/Salida que realice la comunicación entre el Módulo de Capa Física y el Módulo de Protocolos. Este bloque puede ser genérico, válido para cualquier Sistema de Pruebas, pero necesita incorporar

SPICUM servicio de publicaciones

la definición de las primitivas utilizadas. Sin embargo, este bloque puede tener que ser implementado en cada ocasión ya que depende de un elemento, el Módulo de Capa Física, que puede haber sido desarrollado por un tercero. En este caso, mantener el mismo mecanismo de comunicación en distintos Sistemas de Pruebas puede ser inviable. De igual forma, hay que construir el bloque codificador para la comunicación entre ambos módulos, Codec $Local_{PHY}$. Este codificador va unido a cómo se haya definido la comunicación con el Módulo de Capa Física y su reutilización o no depende de los mismos factores.

El Subsistema Inferior se comprueba con las Pruebas de Subsistema. El resultado de las mismas se documenta en el Informe de Pruebas de Subsistema, donde se debe incluir una lista de las pruebas realizadas y los cambios que haya sido necesario realizar en cada modelo. Lo más conveniente para realizar estas pruebas es ejecutar el modelo SDL del Módulo de Protocolos dentro del simulador del entorno de desarrollo, utilizando la opción de tiempo real.

Dependiendo de la funcionalidad incluida en los emuladores de niveles remotos y superiores, el sistema formado por la integración de éstos con el Módulo de Capa Física se puede considerar un emulador de la Implementación Bajo Prueba. Este emulador se puede utilizar para realizar las Pruebas de Sistema en la última fase del diseño.

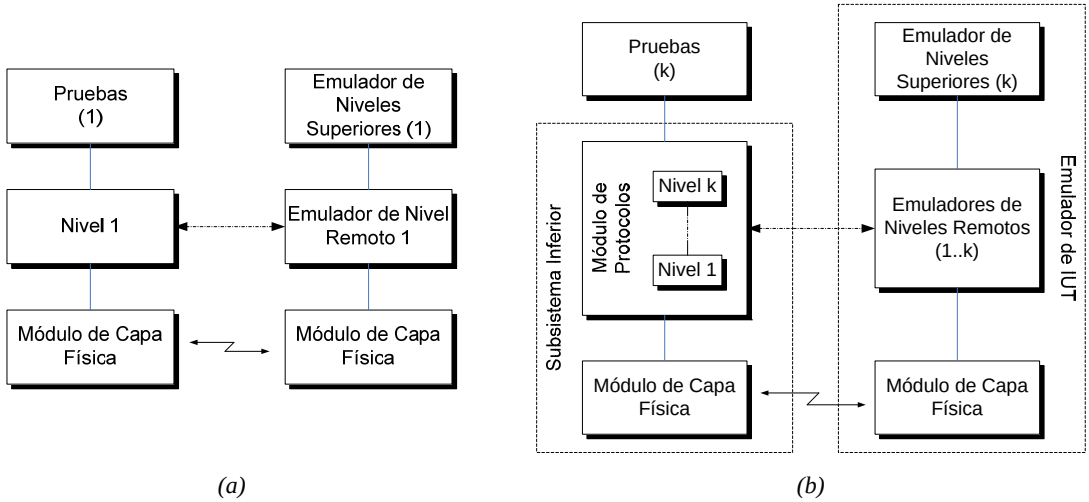


Figura 4.39: Pasos en la integración del Módulo de Capa Física con el Módulo de Protocolos.

4.5.3.2.1 Entradas y Salidas

La subfase Integración del Módulo de Protocolos y el Módulo de Capa Física está pensada para comprobar que la integración de los distintos elementos que constituyen el Subsistema Inferior funcione correctamente.

SPICUM servicio de publicaciones

Tabla 4.18: Entradas y salidas de la subfase Integración del Módulo de Capa Física.

Entradas	Salidas

• Especificaciones de Sistema
• Arquitectura
• Módulo de Protocolos
  o Estructura de bloques
  o Modelos SDL de niveles
  o Integrado
  o Interfaces
  ▪ Módulo de Capa Física
  ▪ Adicional de Control
  ▪ (opc) Subsistema de Pruebas
• Módulo de Capa Física
• Emuladores
• Plan de Pruebas
  o Subsistema
• Pruebas
  o (opc) Pruebas Propias
  o (opc) Juegos de Pruebas
	• Subsistema Inferior
	  o Sin adaptación a la plataforma
	  o Módulo de Protocolos y Módulo de Capa Física integrados
	• (opc) Gestión de Entrada/Salida
	  o Módulo de Capa Física
	  ▪ Codec $Local_{PHY}$
	• (opc) Emulador de IUT
	• Informes de Pruebas
	  o Subsistema

Herramientas	

• Entorno de desarrollo en SDL
  o Editor
  o Analizador sintáctico y semántico
  o Generador de código
  o Simulador
• (opc) Entorno de desarrollo en TTCN
  o Simulador
• Entorno de desarrollo en C
  o Editor
  o Compilador
  o Depurador
• Editor de MSCs	

4.5.3.3 Obtención de una Entidad Ejecutable del Subsistema Inferior

La generación de un ejecutable del Subsistema Inferior es muy similar a la generación de un ejecutable del Subsistema de Pruebas. Los pasos a realizar son los siguientes:

1. Generación de Código: Al igual que la notación TTCN, SDL no está pensado para se ejecutado directamente. Las herramientas actuales generan código C o C++ a partir del modelo SDL. Este código se puede utilizar con los compiladores típicos (gcc, Borland C++, MS Visual C++), tanto comerciales como de libre distribución, y puede ejecutarse en la mayoría de plataformas. Se ha decidido emplear código C.
2. Construcción de Componentes:

SPICUM servicio de publicaciones

a. Gestión de Entrada/Salida: El bloque de Gestión de Entrada/Salida tiene dos interfaces:

   i. Comunicación con el Subsistema de Pruebas: La gestión de esta interfaz es genérica, pero requiere ser adaptada con la definición de las primitivas empleadas por cada Módulo de Protocolos.

   ii. Comunicación con el Módulo de Capa Física: Al igual que en el caso anterior, esta interfaz puede ser genérica, pero necesita, al menos, incorporar la definición de las primitivas utilizadas. Como se ha comentado en el apartado anterior, esta interfaz es susceptible de tener que ser implementada en cada ocasión.

b. Codificadores:

   i. Codec $Local_{TTCN}$: Este codificador se puede generar automáticamente mediante la herramienta *GenCod* (Capítulo 5, Sección 5.5.2.2), a partir de la salida generada por *GenDef*.

   ii. Codec $Local_{PHY}$: Este codificador va unido a como se haya definido la comunicación con el Módulo de Capa Física y su reutilización o no depende de los factores indicados previamente.

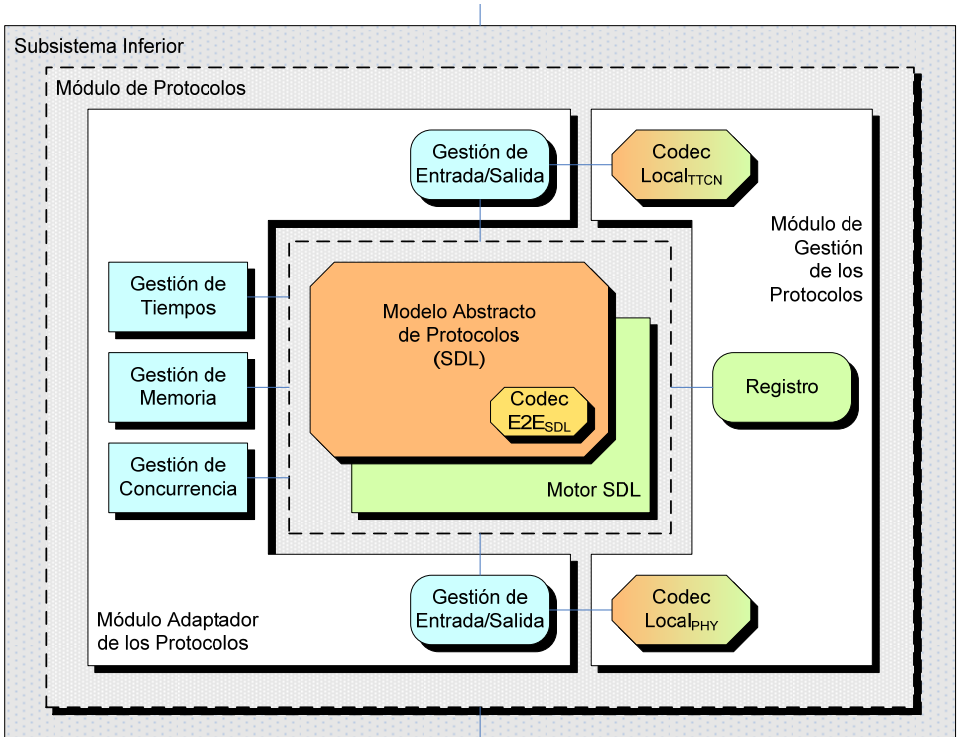


Figura 4.40: Estructura detallada del Módulo de Protocolos.

3. Obtención del Ejecutable: La última tarea es la compilación y enlazado de los componentes del Módulo de Protocolos. Para obtener el ejecutable del Módulo de Protocolos hay que compilar y enlazar los siguientes elementos (Figura 4.40):

   - Código generado en el paso 1
   - Motor SDL

SPICUM servicio de publicaciones

- Módulo Adaptador de los Protocolos
- Módulo de Gestión de los Protocolos

Como resultado se obtiene una entidad ejecutable del Subsistema Inferior que está formada por la integración de un ejecutable del Módulo de Protocolos y un Módulo de Capa Física.

4.5.3.3.1 Entradas y Salidas

Las entradas y salidas de la subfase Obtención de una Entidad Ejecutable del Subsistema Inferior son los componentes abstractos que forman el mismo y los Módulos necesarios para su ejecución en la plataforma elegida.

Tabla 4.19: Entradas y salidas de la subfase Obtención de una Entidad Ejecutable del Subsistema Inferior.

Entradas	**Salidas**

• Arquitectura
• Subsistema Inferior
  o Módulo de Protocolos y Módulo de Capa Física integrados
• Módulo Adaptador de los Protocolos
• Módulo de Gestión de los Protocolos
• Motor SDL
• Gestión de Entrada/Salida
  o Subsistema de Pruebas
  ▪ Codec $Local_{TTCN}$
  o (opc) Módulo de Capa Física[31]
  ▪ Codec $Local_{PHY}$
• Módulo de Protocolos
  o Interfaces
  ▪ Subsistema de Pruebas
  ▪ Adicional de Control
	• Subsistema Inferior
	  o Entidad ejecutable
	• (opc) Gestión de Entrada/Salida[28]
	  o Módulo de Capa Física
	  ▪ Codec $Local_{PHY}$

Herramientas	

• Entorno de desarrollo en SDL
  o Generador de código
• Entorno de desarrollo en C
  o Editor
  o Compilador
	• Generador de Interfaces
	  o Generador de Codificador de la Interfaz

4.5.3.4 Entradas y Salidas

La fase Integración del Subsistema Inferior está pensada para comprobar que la integración de los distintos elementos que constituyen este Subsistema funcione correctamente y obtener una entidad ejecutable del mismo.

[31] El Codec $local_{PHY}$ puede ser una entrada o una salida, según si se dispone previamente de él o hay que realizarlo.

SPICUM servicio de publicaciones

Tabla 4.20: Entradas y salidas de la fase Integración del Subsistema Inferior.

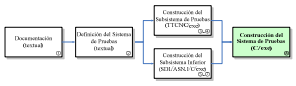
Entradas	Salidas

• Especificaciones de Sistema
• Arquitectura
• Módulo de Protocolos
  o Estructura de bloques
  o Modelos SDL de niveles
  o Interfaces
  ▪ Módulo de Capa Física
  ▪ Adicional de Control
  ▪ Subsistema de Pruebas
• Módulo de Capa Física
• Módulo Adaptador de los Protocolos
• Módulo de Gestión de los Protocolos
• Motor SDL
• Gestión de Entrada/Salida
  o Subsistema de Pruebas
  ▪ Codec $Local_{TTCN}$
  o (opc) Módulo de Capa Física[31]
  ▪ Codec $Local_{PHY}$
• Emuladores
• Plan de Pruebas
  o Subsistema
• Pruebas
  o (opc) Pruebas Propias
  o (opc) Juegos de Pruebas
	• Subsistema Inferior
	  o Entidad ejecutable
	  o (opc) Gestión de Entrada/Salida[31]
	  ▪ Codec $Local_{PHY}$
	• Informes de Pruebas
	  o Módulo
	  o Subsistema
	• (opc) Emulador de IUT

Herramientas	

• Entorno de desarrollo en SDL
  o Editor
  o Analizador sintáctico y semántico
  o Generador de código
  o Simulador
• (opc) Entorno de desarrollo en TTCN
  o Simulador
• Entorno de desarrollo en C
  o Editor
  o Compilador
  o Depurador
• Editor de MSCs
	• Generador de Interfaces
	  o Generador de Codificador de la Interfaz

4.6 Construcción del Sistema de Pruebas



La última fase de la Metodología es la Construcción del Sistema de Pruebas. El Sistema de Pruebas está constituido por tres Subsistemas (Figura 4.41):

- Subsistema de Operación y Administración
- Subsistema de Pruebas
- Subsistema Inferior

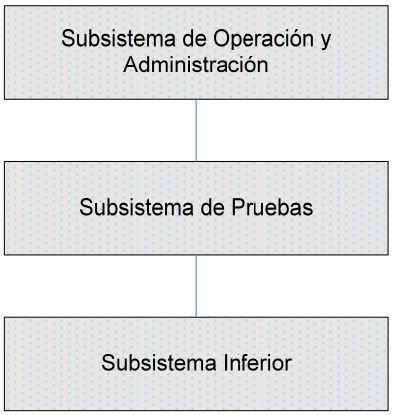


Figura 4.41: Subsistemas que constituyen el Sistema de Pruebas.

El Subsistema de Operación y Administración es un elemento genérico común para todos los Sistemas de Pruebas. Este Subsistema tan sólo necesita ser configurado para que utilice las combinaciones adecuadas de Subsistema de Pruebas y Subsistema Inferior, según las pruebas que se quieran realizar. Se le deben indicar también los Formularios de Parámetros de Pruebas que se deben rellenar antes y después de la realización de las pruebas. Los otros dos Subsistemas, Subsistema de Pruebas y Subsistema Inferior, han sido construidos en las fases anteriores. Cada Subsistema se ve como una entidad ejecutable independiente de los otros Subsistemas.

La comprobación del correcto funcionamiento del Sistema de Pruebas se realiza mediante las Pruebas de Sistema definidas en el Plan de Pruebas. El resultado de estas pruebas se documenta en el correspondiente Informe de Pruebas de Sistema.

Las Pruebas de Sistema se deben realizar empleando equipos comerciales como Implementación Bajo Prueba. Sin embargo, puede que estos equipos no estén disponibles. En este caso, se utilizaría un emulador de Implementación Bajo Prueba, que puede haberse construido en la fase anterior o construirse en esta fase a partir de los desarrollos realizados en las fases anteriores. Este emulador se puede ejecutar en el simulador del entorno de desarrollo utilizando un motor de tiempo real; de esta forma, se reduce su coste de desarrollo.

Al ser esta la última fase de desarrollo, y tratándose de un producto comercial, se debe documentar su utilización en un Manual de Usuario del Sistema de Pruebas.

4.6.1.1 Entradas y Salidas

La fase Construcción del Sistema de Pruebas tiene como objetivo la unión de los tres Subsistemas que lo componen y la verificación de esta integración a través de las Pruebas de Sistema.

SPICUM servicio de publicaciones

Tabla 4.21: Entradas y salidas de la fase Construcción del Sistema de Pruebas.

Entradas	Salidas

• Arquitectura
• Subsistema de Pruebas
• Subsistema Inferior
• Subsistema de Operación y Administración
• Formularios
  o Parámetros de Pruebas (PICS, PIXIT)
• (opc) Emulador de IUT[32]
• Plan de Pruebas
  o Sistema
	• Sistema de Pruebas
	• Manual de Usuario del Sistema de Pruebas
	• (opc) Emulador de IUT[32]
	• Informes de Pruebas
	  o Sistema

Herramientas	

• (opc) Entorno de desarrollo en SDL
  o Simulador
• Entorno de desarrollo en C
  o Depurador
• IUT comercial	

4.7 Planificación Temporal del Diseño

A la hora de plantearse un nuevo proyecto, disponer de una estimación del esfuerzo requerido por cada actividad es muy interesante. Por este motivo se incluyen aquí algunas indicaciones sobre la planificación del diseño. Una estimación temporal se muestra en el diagrama de Gantt [ADAM31] de la Figura 4.42, donde la duración de cada tarea se ha valorado a partir de la experiencia obtenida durante la realización de esta Tesis. Este gráfico permite obtener una idea acerca de la complejidad proporcional de cada tarea con respecto a las otras. Además, muestra qué tareas pueden realizarse simultáneamente, permitiendo distribuir los recursos de forma adecuada.

Se ha considerado que los ingenieros que realizan el diseño tienen competencias diversas, y se han clasificado en ingenieros con a) competencias en TTCN, b) competencias en SDL y c) competencias en hardware. A la mayoría de las fases se les han asignado dos ingenieros a tiempo completo con las competencias necesarias, aunque algunas actividades deben ser realizadas por todos ellos, en común o individualmente, como, por ejemplo, la definición del Sistema de Pruebas. En otras fases, como en el caso de la integración del Subsistema Inferior, se espera la colaboración de los responsables de las entradas utilizadas para la realización de las correspondientes pruebas y el análisis de los resultados; sobre todo, si se detectan errores.

[32] El uso del emulador de IUT es opcional, dependiendo de cómo se realicen las Pruebas de Sistema. Si se usa, puede ser una entrada de esta fase, pero si no se dispone previamente de él, será una salida.

SPICUM servicio de publicaciones

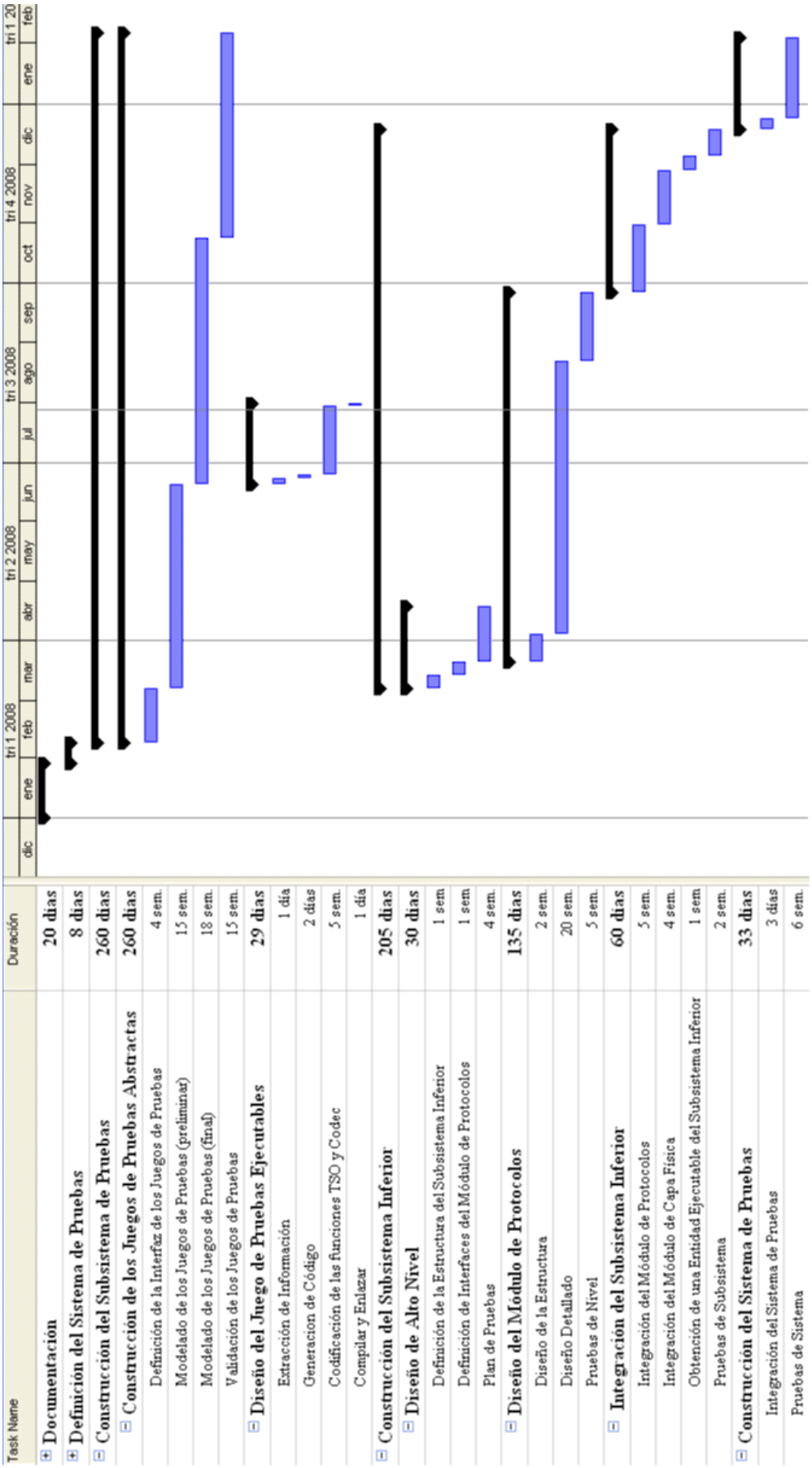


Figura 4.42: Diagrama de Gantt con una posible planificación de las fases de la Metodología de Diseño.

SPICUM
servicio de publicaciones

La construcción de los Juegos de Pruebas Abstractas requiere un comentario aparte, ya que conlleva una alta carga de trabajo, tanto en el modelado como en la validación. Dado que el interés de la planificación es la construcción del Sistema de Pruebas, se ha asignado un número suficiente de ingenieros, en concreto, cuatro, para que esta actividad no impacte la duración total del proyecto. Un número mayor de personas podrían reducir en algo la duración de esta actividad, pero no significativamente, ya que la parte inicial del trabajo requiere un modelado coherente de los escenarios de pruebas, algo difícil de conseguir con un grupo numeroso.

La Metodología permite realizar de forma simultánea la construcción del Subsistema de Pruebas y del Subsistema Inferior, lo que se refleja en la planificación. No obstante, la construcción del Subsistema Inferior requiere un trabajo previo en el Subsistema de Pruebas, por lo que su diseño se inicia más tarde. Se ha considerado que antes de disponer de los Juegos de Pruebas completos se dispone de una versión preliminar de los mismos con el modelado de un subconjunto de pruebas. Esto hace que se pueda obtener un Subsistema de Pruebas suficientemente temprano, aunque será necesario reconstruir este componente con las sucesivas versiones de los Juegos de Pruebas.

En total, en esta estimación, la construcción de un Sistema de Pruebas requiere unas 58 semanas naturales. Dejando un 20% del tiempo para otras actividades, el tiempo estimado puede quedar en unas 70 semanas, es decir, unos 18 ~ 20 meses. La planificación mostrada asume que los elementos genéricos del Sistema de Pruebas están a disposición del equipo de desarrollo y, por tanto, su coste es cero. Pero no considera el mantenimiento del Sistema de Pruebas por la detección de errores en el mismo ni por modificaciones en los Juegos de Pruebas. La obtención de un producto plenamente comercial puede necesitar de otras actividades que no han sido tenidas en cuenta en esta planificación.

4.8 Conclusiones

En este capítulo se ha descrito una Metodología de Diseño de Sistemas de Pruebas, basada en la Arquitectura presentada en el capítulo anterior. Esta Metodología se ha estructurado en ocho fases, para cada una de las cuales se han descrito las actividades que la componen, indicando tanto las entradas de las que parte como las salidas que proporciona. Así, la Metodología se puede ver como un flujo de modelos que son transformados por las distintas actividades. El proceso de diseño se puede realizar de forma iterativa incrementando la funcionalidad en cada ciclo.

La Metodología parte de las Especificaciones de Sistema y Especificaciones de Prueba y, a partir de ahí, engloba todas las actividades necesarias para el diseño de un Sistema de Pruebas. De entre todas las actividades, el diseño del Módulo de Protocolos y el diseño del Juego de Pruebas Abstractas son las más costosas. La utilización de lenguajes formales como parte de la Metodología es un elemento que puede verse como una rémora en las etapas iniciales del proyecto; sin embargo, este coste inicial queda compensado por los beneficios que proporcionan en las etapas posteriores. Se ha presentado una planificación típica, basada en la experiencia obtenida, que permite aprehender la complejidad de cada actividad en relación con las demás y puede servir de base para la planificación de otros diseños.

La Metodología ha sido pensada para realizar un diseño eficiente, para lo que utiliza tanto herramientas comerciales como herramientas propias. Aunque la Metodología es independiente de entornos de desarrollo específicos, se espera que estos proporcionen

SPICUM servicio de publicaciones

funcionalidades para la edición, simulación y depuración de los modelos generados. Las herramientas propias de que hace uso de la Metodología serán descritas en el próximo capítulo.

SPICUM servicio de publicaciones

SPICUM
servicio de publicaciones

Capítulo

5

"Sin embargo yo pregunté los nombres de las piezas y pasé mucho tiempo allí, y mi Fyunch(click) pareció sorprenderse mucho. Pude interpretar mal la emoción, desde luego, pero creo realmente que mi interés por las herramientas les desconcertó."

Hardy, La Paja en el Ojo de Dios

CAPÍTULO 5: HERRAMIENTAS DE SOPORTE

El término herramienta proviene del latín *ferramenta* y se refiere a "un dispositivo o pieza de equipo que posibilita o proporciona una ventaja mecánica para realizar una tarea" [TOOL07]. En términos software, una herramienta es "un programa utilizado principalmente para crear, manipular, modificar, o analizar otros programas" [CATB07].

Los entornos comerciales de desarrollo ofrecen un conjunto de soluciones generalmente adaptado a las necesidades comunes del mercado de usuarios a los que se orientan. A la hora de utilizar estos entornos por parte de un usuario concreto, lo habitual es que ciertas funcionalidades de interés no existan en el entorno elegido. Mientras los componentes centrales de la arquitectura de Sistemas de Pruebas encuentran satisfacción en el entorno de desarrollo, la labor de integración de las diferentes partes así como la interacción con el usuario y la adaptación con la plataforma muestran carencias que es posible cubrir con herramientas propias.

La funcionalidad que debe proporcionar cada una de estas herramientas está en consonancia con la Metodología presentada en el capítulo anterior; sin embargo, la implementación concreta de esta funcionalidad depende del entorno de desarrollo utilizado. Por ello, en primer lugar, y con objeto de explicar esta implementación, la sección 5.1 justifica la elección de un entorno de desarrollo específico.

Las herramientas de soporte a la Metodología de Diseño (Tabla 5.1) descritas en este capítulo son de dos tipos: componentes software genéricos, que son utilizados en unión con otros componentes, y generadores automáticos, que transforman información en componentes software. Estas herramientas se aplican a los componentes de la Arquitectura resaltados en la Figura 5.1. Se pueden dividir en cuatro grupos:

1) Subsistema de Operación y Administración (sección 5.2): Ofrece la posibilidad de que un operador controle la ejecución de las diversas pruebas. Además, realiza otras funciones adicionales como la selección de pruebas, la generación de informes, el análisis de los resultados, etc.

SPICUM servicio de publicaciones

Tabla 5.1: Herramientas de soporte a la Metodología de Diseño.

	Nombre	Utilidad
	Subsistema de Operación y Administración	Interacciona con el operador y permite la ejecución y el control de los Casos de Prueba sobre la Implementación Bajo Prueba, así como la visualización y gestión de los resultados.
Componentes Genéricos	*Subsistema de Pruebas*	
	Módulo Adaptador de las Pruebas	Ofrece acceso a los servicios de bajo nivel de la plataforma de ejecución.
	Módulo de Gestión de las Pruebas	Proporciona servicios de alto nivel para la ejecución de los Casos de Prueba, el registro de la ejecución y la comunicación con otros Subsistemas.
	Subsistema Inferior	
	Módulo Adaptador de los Protocolos	Ofrece acceso a los servicios de bajo nivel de la plataforma de ejecución.
	Módulo de Gestión de los Protocolos	Proporciona servicios de codificación y el registro de la ejecución.
Generadores Automáticos	*Generador de Interfaces (GenInt)*	
	Generador de la Definición de la Interfaz (*GenDef*)	Extrae la definición de la interfaz de los Juegos de Pruebas y la convierte a una sintaxis válida en un modelo SDL.
	Generador de Codificador de la Interfaz (*GenCod*)	Genera las funciones de codificación y descodificación necesarias para la comunicación entre el Subsistema de Pruebas y el Subsistema Inferior.
	Generador de Codificador/Descodificador para la Interfaz Aire (*GenCodecAir*)	Genera las funciones de codificación y descodificación para transmitir por la interfaz aire un conjunto de mensajes de un nivel determinado.
	Reglas de Nombrado	Reglas para nombrar los distintos elementos de un sistema SDL con objeto de evitar duplicidades.

2) Componentes genéricos de los Subsistemas de Pruebas e Inferior: Conjuntos de funciones requeridos por el código generado a partir de los módulos SDL y TTCN. El entorno de desarrollo proporciona la cabecera de muchas de estas funciones, pero no su implementación. Parte de estas herramientas son comunes a todo Sistema de Pruebas, mientras que otra parte se genera de forma semi-automática.
   a. Subsistema de Pruebas (sección 5.3):
      i. Módulo Adaptador de las Pruebas (sección 5.3.2): adaptación a la plataforma del código generado a partir de los Juegos de Pruebas.
      ii. Módulo de Gestión de las Pruebas (sección 5.3.3): funciones de control, registro de trazas y codificación de interfaces.
   b. Subsistema Inferior (sección 5.4):
      i. Módulo Adaptador de los Protocolos (sección 5.4.1): Módulo de adaptación para ejecutar un modelo SDL en una plataforma específica.

SPICUM servicio de publicaciones

ii. Módulo de Gestión de los Protocolos (sección 5.4.2): funciones registro de trazas y codificación de interfaces.

3) Generadores automáticos: Herramientas para generar interfaces y para codificar y descodificar mensajes. Hay dos categorías:

   a. Generador de Interfaces (*GenInt*) (sección 5.5.2): Esta herramienta ayuda en la definición de la interfaz entre el Subsistema de Pruebas y el Subsistema Inferior y en su implementación. Está formado por dos utilidades:

      i. Generador de la Definición de la Interfaz (*GenDef*) (sección 5.5.2.1): Extrae la definición de la interfaz de los Juegos de Pruebas y la convierte a una sintaxis válida en un modelo SDL.

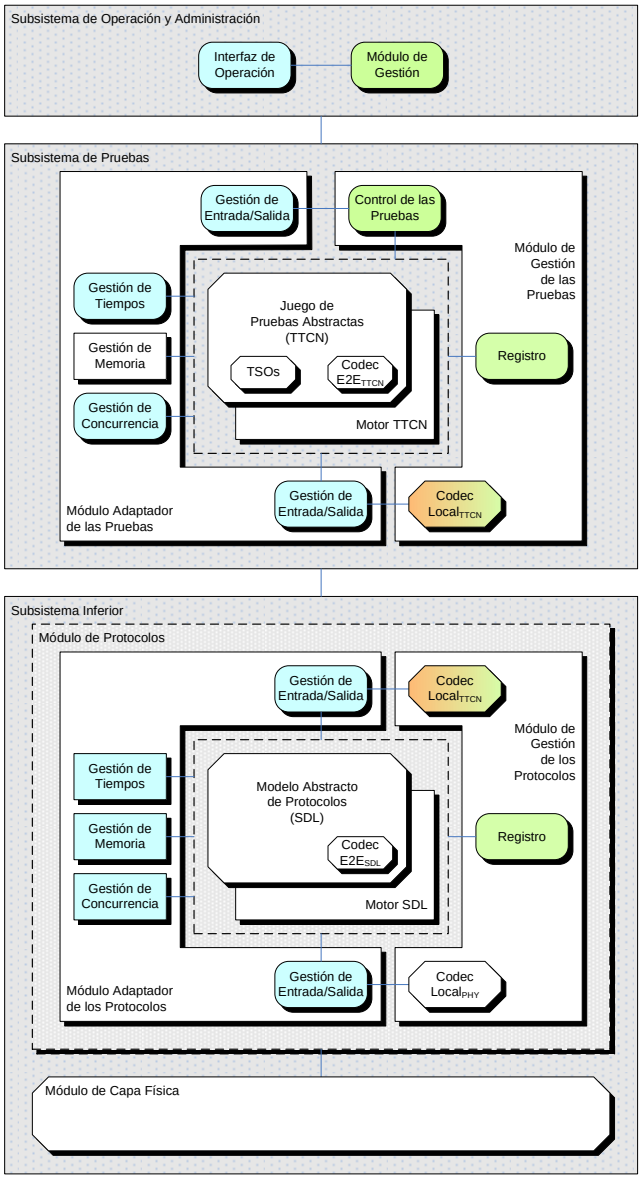


Figura 5.1: Componentes de la arquitectura relacionados con las herramientas descritas en este capítulo.

ii. Generador de Codificador de la Interfaz (*GenCod*) (sección 5.5.2.2): A partir de la definición de la interfaz genera las funciones de codificación

SPICUM servicio de publicaciones

y descodificación necesarias para la comunicación entre los dos módulos.

b. Generador de Codificador/Descodificador para la Interfaz Aire (*GenCodecAir*) (sección 5.5.3): Genera las funciones de codificación y descodificación para transmitir por la interfaz aire un conjunto de mensajes de un nivel determinado.

4) Reglas de Nombrado (sección 0): Reglas para nombrar los distintos elementos de un sistema SDL con objeto de evitar duplicidades.

Este capítulo se ha estructurado de la siguiente forma. En primer lugar, se comentan brevemente las herramientas comerciales utilizadas para las implementaciones de Sistemas de Pruebas descritas en los capítulos 7 a 9. El resto del capítulo está dedicado a las herramientas diseñadas. La Sección 5.2 describe el Subsistema de Operación y Administración. Las Secciones 5.3 y 5.4 presentan los componentes genéricos del Subsistema de Pruebas y el Subsistema Inferior. La Sección 5.5 explica cómo se han implementado los generadores automáticos. Por último, se exponen las reglas de nombrado que se han definido.

5.1 Entornos de Desarrollo

La Metodología de Diseño está basada en el uso de los lenguajes formales SDL y TTCN, lenguajes para los que existe una escasa variedad de entornos de desarrollo comerciales[1]. Y la mayoría de estas herramientas se limitan al ámbito de uno de los dos lenguajes. Para SDL se dispone de las herramientas *Cinderella* [CINDER], *ezSDL* [EZSDL], *RT Developer Studio* [RTDEVE], *Safire-SDL* [SAFIRE] y *Tau Suite* [TAU]. Todas estas herramientas incluyen la funcionalidad de edición, análisis, simulación y generación de código; además, la mayoría incorporan un visor o un editor del lenguaje MSC. En el lado de TTCN, las herramientas *OpenTTCN* [OPENTT], *Tau Suite* [TAU] y *TThree* [TTHREE] permiten editar, analizar, simular y generar código; las herramientas *Acacia* [ACACIA] y *Danet* [DANET] son solamente generadores de código, y utilizan la herramienta *Leonardo* [LEONAR] como visor y/o editor.

Al iniciar este trabajo de investigación el único entorno de desarrollo que incluía herramientas para ambos lenguajes, SDL y TTCN, era *Tau Suite*, lo que fue determinante para su elección. Con el paso del tiempo, al ir apareciendo nuevas herramientas para el ámbito de SDL se ha planteado la duda de si es razonable mantener la decisión original. La funcionalidad ofrecida por estas herramientas es similar a la de *Tau Suite*, lo que no aportaría ventajas al uso de una herramienta diferente. Se podría considerar si utilizar también una herramienta diferente para TTCN. Para ello, se ha evaluado alguna de estas nuevas herramientas, pero se ha visto que ofrecen una funcionalidad similar y no mejoran las limitaciones de *Tau Suite*. En concreto, se ha evaluado la herramienta *TTCN Toolbox* ([SANC02], [SORE02]) (Capítulo 9, Sección 9.3).

5.1.1 Tau Suite

La primera versión de la herramienta *Tau* aparece en 1988, aunque la integración de herramientas para TTCN no se produce hasta 1995. Actualmente incluye herramientas

[1] En el ámbito académico han sido desarrollados diversos prototipos, aunque suelen estar enfocados a necesidades concretas y carecer de una evolución de su funcionalidad. Igualmente, algunas compañías han desarrollado internamente sus propias herramientas, aunque con la evolución y maduración de las herramientas comerciales, estos desarrollos internos han ido siendo abandonados.

SPICUM servicio de publicaciones

para el desarrollo con SDL, MSC, TTCN, ASN.1 y UML; además, permite incluir código C en modelos SDL. Dispone de editores, analizadores sintácticos y semánticos, simuladores, generadores de código y una miríada de utilidades adicionales. Entre ellas se encuentran diversos núcleos de adaptación que permiten utilizar el código generado en un entorno de simulación o uno de ejecución, en tiempo real o no, según las necesidades, una utilidad para el análisis del porcentaje de código utilizado en una ejecución o la posibilidad de convertir un diagrama MSC a un sistema SDL que implemente el comportamiento descrito en el diagrama.

El código que genera la herramienta a partir de un modelo SDL se puede portar a multitud de plataformas, como por ejemplo, Solaris, Linux, Windows, RTOS, VxWorks, Lynx, QNX, etc. Se ha comprobado esta posibilidad en los tres primeros sistemas operativos y en DSP/BIOS, que es el sistema operativo de Texas Instruments para los DSPs de la familia TMS320.

5.1.2 TTCN Toolbox

La herramienta *TTCN Toolbox* es un producto de Danet Group [DANET], que proporciona un entorno de desarrollo completo para TTCN: editor, analizador sintáctico y semántico, simulador, generador de código. Incluye un gestor de campañas de pruebas que permite la ejecución de las mismas. El código generado se puede portar también a distintas plataformas (Linux, Solaris, Windows). A diferencia de la herramienta anterior, para cada Caso de Prueba de un Juego de Pruebas, esta herramienta genera un fichero en código C/C++ independiente.

5.2 Subsistema de Operación y Administración

El Subsistema de Operación y Administración es la interfaz a través de la cual el operador del Sistema de Pruebas puede controlar la ejecución de los Casos de Prueba y observar sus resultados. Esta herramienta se ha construido en Java [JAVA], lo que permite su utilización en las plataformas más difundidas (Linux, UNIX, Windows). El desarrollo se ha realizado con el entorno JDK (*Java Development Kit*) proporcionado por Sun; su ejecución requiere la instalación de la máquina virtual (JRE – *Java Runtime Environment*) para la correspondiente plataforma. Tanto el entorno como la máquina virtual son de libre uso y se pueden descargar de la página oficial de Sun Microsystems para Java [JDK].

Para el diseño de la interfaz gráfica (Figura 5.3) se ha utilizado la librería Swing [SWING], que ofrece los componentes típicos a los que un usuario está acostumbrado: barra de menús, menús emergentes, diálogos, botones, selector de alternativas, barras de desplazamiento, tablas, árboles, etc. Swing sustituyó a la librería AWT (*Abstract Windows Toolkit*) aportando mayor flexibilidad y un mayor número de componentes gráficos.

Internamente, el Subsistema de Operación y Administración está formado por los módulos mostrados en la Figura 5.2 [FERR00]:

a) Ventana Principal: permite al usuario acceder al Sistema de Pruebas.

b) Recursos Gráficos: elementos gráficos necesarios para la herramienta.

SPICUM servicio de publicaciones

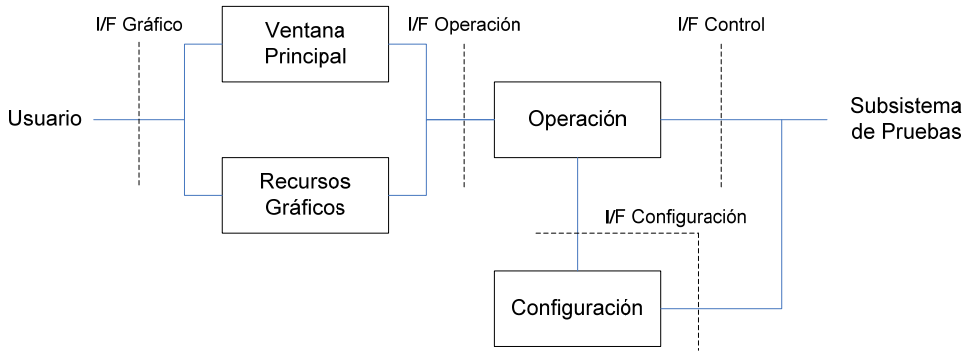


Figura 5.2: Módulos software que forman el Subsistema de Operación y Administración.

c) Operación: responde a las opciones del menú principal, ejecutando los comandos correspondientes. Cada comando se ejecuta como una hebra diferente, lo que evita el bloqueo de la ventana principal.

d) Configuración: acceso a los ficheros de configuración de la herramienta y los asociados al Subsistema de Pruebas.

La ventana principal consta de una barra de menú y una zona de mensajes donde se van imprimiendo aquellos mensajes destinados al operador e información relacionada con la ejecución de las Pruebas. El menú principal incluye cuatro submenús, cuyas opciones se desactivan si no es posible realizar la acción. La funcionalidad a que se accede a través de cada submenú es la siguiente:

a) Submenú *`Casos`*: Seleccionar, modificar, ejecutar, detener y abortar Pruebas. La opción *`Seleccionar`* permite elegir el Subsistema de Pruebas y crear una selección (Figura 5.4) de entre los Casos de Prueba incluidos en el mismo. La opción *`Modificar`* permite modificar el conjunto de Casos de Prueba seleccionado; sólo se activa si ya existe una selección. El resto de opciones se explica por sí mismo[2].

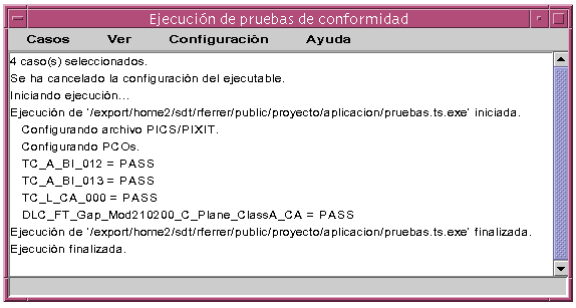


Figura 5.3: Ventana principal del Subsistema de Operación y Administración.

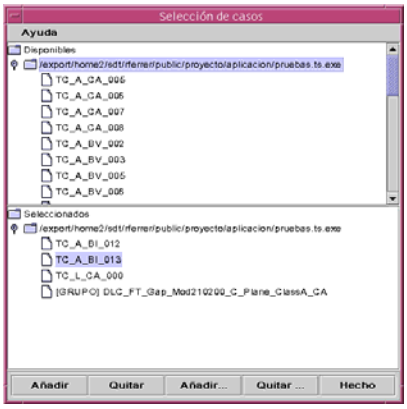


Figura 5.4: Selector de Casos de Prueba.

[2] La opción *`Detener`* permite una parada controlada de la ejecución de la serie de Pruebas en curso, deteniendo la misma sólo tras la finalización de un Caso de Prueba; la opción *`Abortar`* sólo debe usarse si la ejecución se ha quedado bloqueada.

SPICUM servicio de publicaciones

b) Submenú *Ver*: Acceso al visor de trazas de las ejecuciones realizadas y al editor de los Parámetros de Pruebas.

c) Submenú *Configuración*: Acceso a los parámetros de configuración del Subsistema de Operación y Administración (directorios de trabajo y conjunto de señales de respuesta del operador[3]) y a la selección de los ficheros asociados con el Subsistema de Pruebas (ejecutable del Subsistema Inferior y fichero de Parámetros de Pruebas).

d) Submenú *Ayuda*: Información sobre el uso de la herramienta.

Tabla 5.2: Acceso a la funcionalidad básica del Subsistema de Operación y Administración.

Funcionalidad	**Acceso a través de ...**
Edición y visualización de los Parámetros de Pruebas.	Submenú *Ver* | *PICS/PIXIT*
Selección del Subsistema de Pruebas.	Submenú *Casos* | *Seleccionar*
Identificación de los Casos y Grupos de Pruebas aplicables a un equipo, en función de los Parámetros de Pruebas.	Submenú *Casos* | *Seleccionar*
Selección de uno o más Casos de Prueba.	Submenú *Casos* | *Seleccionar*
Ejecución de los Casos de Prueba seleccionados.	Submenú *Casos* | *Ejecutar*
Registro de los veredictos de la ejecución y de las trazas generadas durante la misma.	Registro en tiempo de ejecución. Veredicto mostrado en pantalla.
Posibilidad de abortar la ejecución de la Prueba o las Pruebas cuya ejecución se encuentre en curso.	Submenú *Casos* | *Abortar* Submenú *Casos* | *Detener*
Visualización textual y gráfica de las trazas generadas durante las ejecuciones (una o más) de cada Prueba.	Modo texto: Submenú *Ver* | *Trazas* Modo gráfico: Visor de Trazas, submenú *MSC* | *Ver MSC*
Filtrado de las trazas en función de su contenido.	Visor de Trazas, submenú *Presets*
Volcado de las trazas a un fichero con formato MSC.	Visor de Trazas, submenú *MSC* | *Exportar*

En el Capítulo 3, Sección 3.2.1, se definió la funcionalidad básica que el Subsistema de Operación y Administración debía incluir; la Tabla 5.2 muestra cómo es posible acceder a esta funcionalidad dentro de la herramienta.

La secuencia de pasos que realiza el Subsistema de Operación y Administración para ejecutar una Selección de Pruebas se refleja en la Figura 5.5. Primero se inicia el Subsistema Inferior y después se inicia el Subsistema de Pruebas; la conexión entre ambos se realiza de forma automática. Después, se configuran las Pruebas. Durante la ejecución de un Caso de Prueba, se pueden recibir comandos desde el Módulo de Gestión de las Pruebas, incluyendo los mensajes de trazas que se generan. Además, se genera un registro de todas las ejecuciones de cada Caso de Prueba, asociando a cada una las trazas correspondientes.

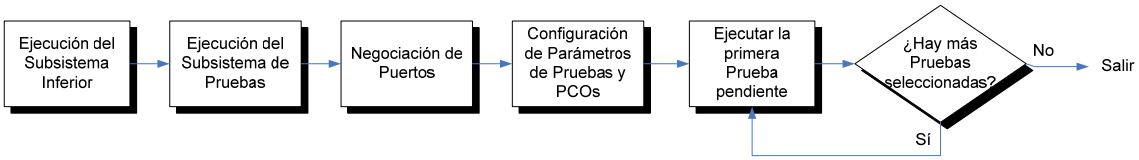


Figura 5.5: Fases en la ejecución de una Selección de Casos de Prueba.

[3] Se trata de señales relacionadas con envíos implícitos, es decir, aparecen en los Casos de Prueba pero invocan alguna acción del operador. Por ejemplo, encender la IUT. Estas señales se envían mediante la función `GciImplicitSend` en lugar de la función `GciSend` (ver apartado 5.3.2.1.1).

SPICUM servicio de publicaciones

A pesar de la sencillez de la herramienta, dispone de toda la funcionalidad necesaria para un Subsistema de Operación y Administración, con la ventaja de que puede ejecutarse en prácticamente cualquier plataforma. Los desarrollos comerciales pueden ofrecer entornos gráficos más atractivos, así como funciones adicionales que faciliten la operación, pero manteniendo la funcionalidad básica.

A continuación se describen dos utilidades incluidas en el Subsistema de Operación y Administración: el Visor de Trazas y el Editor de Parámetros de Pruebas.

5.2.1 Visor de Trazas

El Visor de Trazas permite analizar la ejecución de una Prueba ya terminada tanto en formato textual como en formato gráfico. La ventana (Figura 5.6) consta de tres zonas:

a) La zona superior izquierda permite seleccionar el Caso de Prueba.

b) La zona superior derecha permite elegir una de todas las ejecuciones anteriores del Caso de Prueba. Para ello, indica la fecha y hora de cada ejecución, junto con un comentario opcional.

c) La parte inferior muestra las trazas generadas durante la ejecución seleccionada.

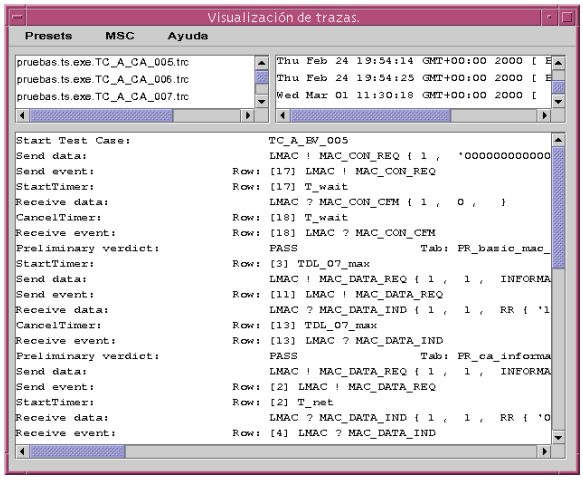


Figura 5.6: Ventana principal del Visor de Trazas.

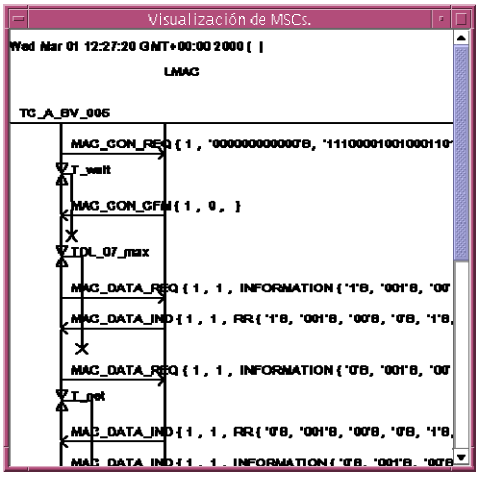


Figura 5.7: Ejemplo de traza representada con un diagrama MSC.

Para facilitar la labor del operador se ha incluido la posibilidad de filtrar las trazas. Se puede elegir entre mostrar todas las trazas, mostrar grupos de trazas predefinidos (por ejemplo, sólo eventos) o personalizar qué tipos de trazas se muestran y cuáles se ocultan[4].

Se puede también visualizar la traza escogida mediante un diagrama de secuencia de mensajes (Figura 5.7). Esta representación muestra una entidad por cada PCO del Caso de Prueba más una entidad adicional que representa el Subsistema Inferior; para cada primitiva muestra los valores de sus parámetros. El diagrama resultante se genera a partir de las trazas correspondientes a los siguientes eventos: *Start Test Case, Send Data*, *Receive Data*, *Final Verdict*, *StartTimer*, *StopTimer*, *CancelTimer* y *Timed out*

[4] Los tipos de posibles trazas se explican en la sección 5.3.3.1.2.

SPICUM servicio de publicaciones

Timer. Finalmente, la herramienta permite exportar una traza a un fichero con formato MSC textual, según se define en [Z.120].

5.2.2 Editor de Parámetros de Pruebas

El Subsistema de Operación y Administración ofrece también la posibilidad de editar los ficheros de Parámetros de Pruebas. Estos ficheros incluyen una línea para cada Parámetro (Figura 5.8); en cada línea hay seis campos, separados por un carácter de tabulación (`0x09`): nombre del parámetro, tipo de datos, valor, definición del tipo, comentario y referencia a la norma. Los últimos tres campos corresponden a comentarios incluidos en la definición del tipo en el Juego de Pruebas y pueden estar vacíos. El editor permite modificar el valor de cada Parámetro; el resto de los campos los muestra pues contienen información útil sobre el significado del Parámetro. También se pueden reordenar alfabéticamente las líneas en función del contenido de cualquier columna.

La interpretación del valor de cada parámetro se realiza al inicializar la ejecución de un conjunto de Casos de Prueba, ya que es en ese momento cuando se carga el fichero de Parámetros de Pruebas. En su versión actual, el editor admite todos los tipos TTCN, tanto simples como estructurados[5].

```
TSPC_cc_support     BOOLEAN      TRUE    /*BOOLEAN*/ /*PICS Item A.12/1*/
       /*Is CC entity supported?*/
TSPC_lce_support    BOOLEAN      TRUE    /*BOOLEAN*/ /*PICS Item  A.12/6*/
       /*Is LCE entity supported?*/
TSPC_llme_support   BOOLEAN      TRUE    /*BOOLEAN*/ /*PICS Item  A.12/7*/
       /*Is LLME entity supported?*/
TSPC_access_rights  BOOLEAN      TRUE    /*BOOLEAN*/ /*PICS Item A.14/16*/
       /*Is MM feature On air subscription registration supported?*/
TSPC_user_auth      BOOLEAN      FALSE  /*BOOLEAN*/  /*PICS Item A.19/4*/
       /*Is MM procedure: Authentication of user supported?*/
TSPX_decimal_ac_value OCT_4      '00000001'O  /*Octet_String[4]*/
       /*PIXIT Question B.8.1*/  /*Value of AC to be used. The AC will be
       entered as maximal 8 decimal digits. The AC to bitstring mapping will be
       done with TSO_cinpt_convert_ac_to_bitstring.*/
TSPX_complete_fixed_id_ari_rpn_value    FIXED_ID_VALUE_TYPE
       '00000001000000100000001100000010000000001'B /*Bit_String[32..40]*/
       /*PIXIT Question B.8.2*/ /*Value of fixed_id to be used in case of ARI +
       RPN*/
TSPX_nr_of_digits_in_cpn  INT_8  3     /*Integer(0..255)*/ /*PIXIT Question
       B.7.5*/ /*In order to facilitate testing, a number of digits less then
       10 is advised. This parameter really indicates the number of CC_INFO
       messages to be expected during call setup*/
TSPX_lce_02  INTEGER      10000  /*INTEGER*/  /*PIXIT Question B.7.7*/
       /*Value of Timer T_P_LCE_02 in milliseconds*/
```

Figura 5.8: Ejemplo de fichero de Parámetros de Pruebas para el Juego de Pruebas del Nivel de Red de DECT.

5.2.3 Evaluación del Rendimiento

Para evaluar la carga de procesamiento que supone el Subsistema de Operación y Administración, se han realizado medidas durante la ejecución de una Prueba haciendo uso y sin hacer uso de esta interfaz gráfica.

[5] El editor de Parámetros de Pruebas no soporta tipos ASN.1.

SPICUM servicio de publicaciones

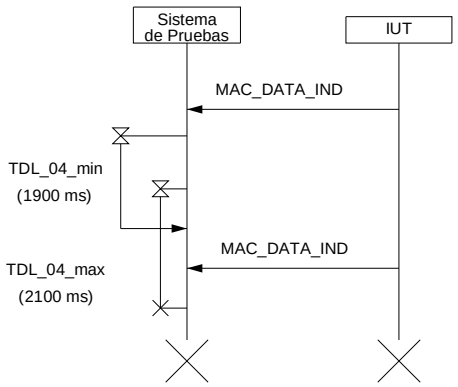


Figura 5.9: Caso de Prueba TC_A_BV_005.

Se ha utilizado la prueba `TC_A_BV_005`, que corresponde al nivel DLC del sistema DECT. La ejecución de este Caso de Prueba tiene los siguientes pasos relevantes (Figura 5.9):

- Recepción de la primitiva `MAC_DATA_IND`.
- Inicio del temporizador `TDL_04_min`, de duración 1900 ms.
- Inicio del temporizador `TDL_04_max`, de duración 2100 ms.
- Vencimiento del temporizador `TDL_04_min`.
- Recepción de la segunda primitiva `MAC_DATA_IND`, aproximadamente 2 segundos después de la primera; es decir, antes de que venza el temporizador `TDL_04_max`. El margen de tolerancia es de ± 100 ms.

Las medidas se han realizado sobre las siguientes plataformas:

- Sun Ultrasparc-1 con Solaris 7.
- Pentium-III 550 MHz con Linux Mandrake 7.0.
- Pentium-III 550 MHz con Windows 98.

Tabla 5.3: Tiempos (en milisegundos) obtenidos en las diferentes plataformas.

	Sin Interfaz Gráfico		**Con Interfaz Gráfico**	
Evento	**Instante**	**Demora**	**Instante**	**Demora**
Recepción primera	0 / 0 / 0	5 / 1 / NA	0 / 0 / 0	5 / 1 / 5
Arranque `TDL_04_min` (1900 ms)	17 / 10 / NA	–	17 / 10 / 49	–
Arranque `TDL_04_max` (2100 ms)	18 / 10 / NA	–	18 / 10 / 64	–
Vencimiento `TDL_04_min`	1918 / 1911 / NA	1 / 1 / NA	1923 / 1911 / 1949	6 / 1 / 0
Recepción segunda	1970 / 1970 / NA	5 / 1 / NA	1969 / 1980 / 1980	5 / 1 / 4

Los tiempos obtenidos en una ejecución típica se muestran en la Tabla 5.3. La columna 'Instante' indica el momento en el cual ha ocurrido el evento; la columna 'Demora' indica el tiempo desde que la señal se encuentra en la cola de recepción hasta que es procesada y, por tanto, tiene efecto. En las celdas donde haya varios valores, el primero corresponde a la plataforma Solaris, el segundo a Linux y el tercero a Windows. Se ha

SPICUM servicio de publicaciones

tomado como instante 0 el momento en el que se recibe la primera señal `MAC_DATA_IND`. Los valores incluidos corresponden a una ejecución media.

En la ejecución sin interfaz gráfico, no se han incluido los valores correspondientes a Windows, pues el comportamiento no ha sido suficientemente estable como para proporcionar unos valores significativos. Los tiempos mostrados en el caso de emplear el interfaz gráfico son algo más estables.

A partir de estas medidas se pueden obtener varias conclusiones:

- Las ejecuciones con interfaz gráfica son menos estables en cuanto a precisión en la plataforma Solaris.
- Hay un retardo significativo desde la primera recepción hasta el arranque de los temporizadores.
- Linux es la plataforma más estable, mientras que Windows no es una plataforma viable para la ejecución de Casos de Prueba con fuertes restricciones temporales si se hace uso de esta herramienta. La tardanza en iniciar los temporizadores puede comprometer la realización de la Prueba (diferencia de tan sólo 31 ms entre el vencimiento del temporizador `TDL_04_min` y la recepción de la segunda primitiva).

Aparte de estas conclusiones, se debe tener en cuenta que una elevada carga de información de traza puede degradar el funcionamiento de esta herramienta. Por dicho motivo, la información detallada sobre el avance de las Pruebas sólo está disponible tras finalizar su ejecución.

5.3 Componentes del Subsistema de Pruebas

El Subsistema de Pruebas requiere de un Módulo Adaptador de las Pruebas y un Módulo de Gestión de las Pruebas para su ejecución. Estos Módulos han sido diseñados para que este Subsistema pueda ejecutarse en las plataformas Linux, Unix y Windows[6]. En primer lugar se describe la Interfaz GCI, empleada por ambos Módulos; a continuación, se describe la implementación de cada uno de ellos.

5.3.1 Interfaz GCI

La Interfaz GCI (*Generic Compiler/Interpreter*) [GCI96] define el conjunto de funciones (ver Apéndice D) que permiten la interacción

a) entre el Juego de Pruebas y los Módulos Adaptador y de Gestión de las Pruebas. En realidad, la interfaz se define para la unión del Juego de Pruebas con el Motor TTCN, denominada Ejecutable de Pruebas o Comportamiento en Tiempo de Ejecución (TTCN-RB – *Runtime Behaviour*).

b) en la frontera superior del Subsistema de Pruebas.

Esta interfaz está dividida en cuatro partes (Figura 5.10):

- Interfaz de Gestión: Mecanismos para inicializar los Casos de Prueba, obtener información sobre los Componentes de Prueba de su configuración y conocer los PCOs y CPs asociados.

[6] Durante la fase de enlazado, para las dos primeras plataformas hay que incluir las librerías `libsocket` y `libnsl`; para la tercera plataforma, hay que definir la opción de compilación `PLATAFORMA_WIN32` e incluir la librería `libwsock32`.

SPICUM servicio de publicaciones

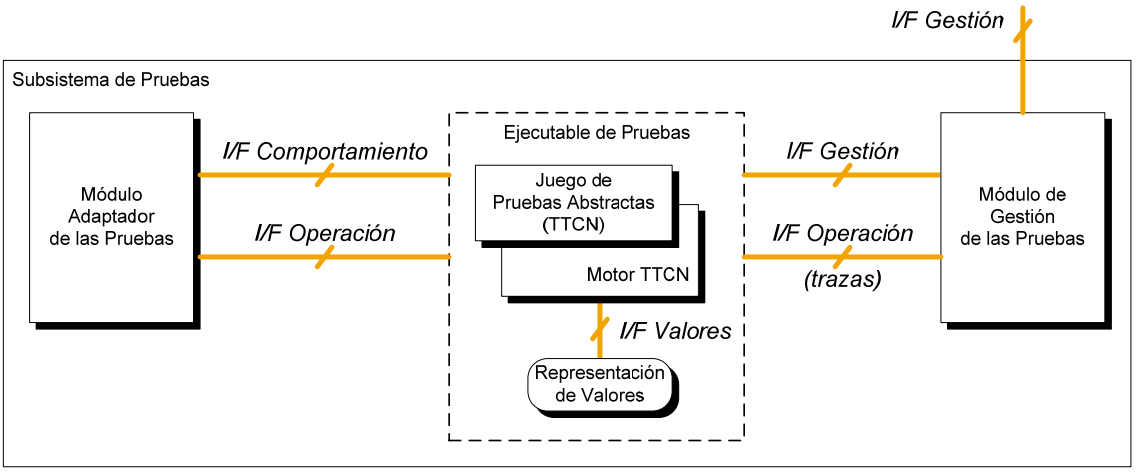


Figura 5.10: Componentes de la Interfaz GCI.

- Interfaz de Operación: Interfaz para enviar señales, manejar temporizadores, generar trazas y crear y configurar Componentes de Prueba.
- Interfaz de Comportamiento: Recepción de señales y vencimientos de temporizadores, e indicación de finalización de Componentes de Prueba.
- Interfaz de Valores: Funciones para obtener y fijar el tipo de un valor (`GciSetType`, `GciGetType`), así como crear y leer valores de un tipo determinado (`GciMk<tipo>`, `GciGet<tipo>`). Las funciones que representan los valores de tipos de datos son un módulo que se puede entender implementado dentro del Motor TTCN, ya que esta representación viene proporcionada por las herramientas de desarrollo.

  Las operaciones de creación y lectura sólo se definen para el caso de un tipo base TTCN[7]. En el caso de tipos estructurados, la interfaz define dos mecanismos alternativos:

  - Gestión flexible de valores: define un conjunto de funciones genéricas que permiten el manejo de tipos estructurados. Referencia los campos de un tipo estructurado mediante un índice numérico.
  - Gestión rígida de valores: define, para cada tipo, una función específica para cada operación que se pueda realizar sobre él o uno de sus campos.

La interfaz GCI define además diversos tipos se utilizan como parámetros y resultados de las funciones de la interfaz. Los más significativos son:

- `GciStatus`: indica si la operación se ha realizado con éxito.
- `GciVerdict`: almacena el veredicto de una Prueba.
- `GciValue`: contenedor que ofrece un tipo genérico para almacenar valores de cualquier tipo. Se corresponde con una unión de todos los tipos posibles.

[7] Los tipos base de TTCN son: `INTEGER`, `BOOLEAN`, `REAL`, `BIT_STRING`, `HEXSTRING`, `OCTET_STRING`, `ENUMERATED`, `CHOICE`, `OBJECT_IDENTIFIER`, `ObjectDescriptor`, `NumericString`, `PrintableString`, `TeletextString`, `VideotexString`, `VisibleString`, `IA5String`, `T61String`, `ISO646String`, `GraphicString`, `GeneralString`, `NULL`, `ANY`, `PDU`, `R_TYPE`.

SPICUM servicio de publicaciones

Las funciones de la interfaz GCI se pueden separar en dos grupos, dependiendo de dónde se implementan. Parte de las funciones se encuentran ya realizadas en el código que genera la herramienta *Tau Suite* y forman parte de las librerías proporcionadas. Básicamente se trata de aquellas funciones que dan información sobre el Juego de Pruebas; el resto de funciones deben ser construidas. La herramienta ofrece una implementación genérica de este segundo grupo de funciones, aunque con algunas limitaciones:

- ✓ Manejo de temporizadores: usa una lista dinámica, pero no es portable a Windows (función `gettimeofday`).
- ✓ Manejo de PCOs: usa descriptores de ficheros no compatibles con una interfaz *socket*, lo que dificulta la portabilidad. Además, el PCO no se abre hasta que se realiza el primer envío, lo que impide enviar mensajes de configuración con antelación.
- ✓ Lectura de parámetros: no se proporciona código.
- ✓ Registro de trazas: usa un descriptor de fichero como salida al igual que con los PCOs.
- ✓ Listado de Casos de Prueba: no se proporciona código.
- ✓ Ejecución de Casos de Prueba: ofrece un ejemplo básico de cómo invocar la ejecución de uno de ellos.
- ✓ Codificación/Descodificación: ofrece la posibilidad de usar una sintaxis de transferencia propia, que denomina ASCII, o reglas de codificación BER o PER. Requiere su adaptación a las necesidades concretas del Sistema de Pruebas.

Debido a estas limitaciones, ha sido necesario implementar o adaptar parte de estas funciones, aunque se ha mantenido en lo posible el conjunto de funciones que define la implementación genérica. Se comentará el diseño en las secciones siguientes.

5.3.2 Módulo Adaptador de las Pruebas

El Módulo Adaptador de las Pruebas implementa la funcionalidad necesaria para que el Juego de Pruebas se pueda ejecutar sobre una determinada plataforma. Se trata de funciones generales, normalmente de bajo nivel, como temporizadores, gestión de colas, gestión de memoria, etc. Se comunica con otros elementos a través de la interfaz GCI (*Generic Compiler/Interpreter*). El Módulo Adaptador de las Pruebas (Figura 5.11) está dividido en cuatro componentes (Capítulo 3, Sección 3.2.2): Gestión de Entrada/Salida, Gestión de Tiempos, Gestión de Memoria y Gestión de Concurrencia.

Además de comunicarse con el Ejecutable de Pruebas, el Módulo Adaptador de las Pruebas se debe comunicar con el Subsistema de Operación y Administración y con el Subsistema Inferior. Por ello, aparte de la interfaz GCI, el Módulo Adaptador de las Pruebas ofrece dos interfaces adicionales:

a) Interfaz de Pruebas (frontera superior): Permite el control de las ejecuciones y la transferencia de mensajes de traza durante las mismas. Los mensajes de esta interfaz se trasladan al Módulo de Gestión de las Pruebas. Esta interfaz permite trabajar a través del Subsistema de Operación y Administración (sec. 5.2) o trabajar en modo consola.

SPICUM servicio de publicaciones

b) Interfaz de Datos (frontera inferior): Permite el intercambio de información entre las Pruebas y el Subsistema Inferior. Esta interfaz define una sintaxis de transferencia, pero debe configurarse según el conjunto particular de primitivas.

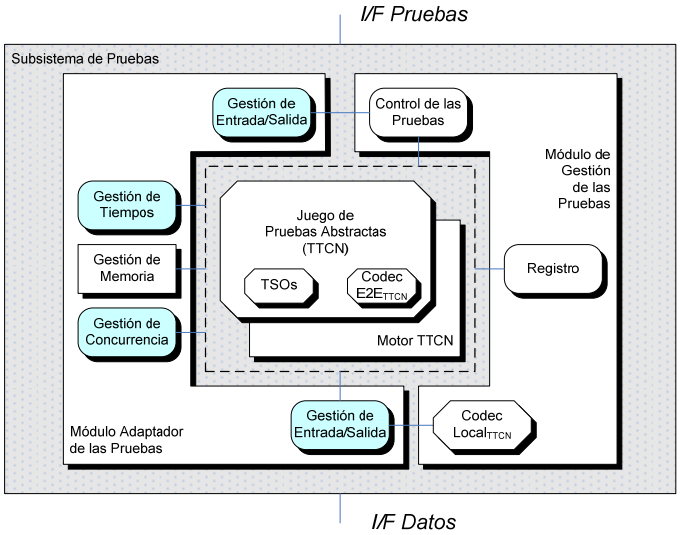


Figura 5.11: Componentes del Módulo Adaptador de las Pruebas.

5.3.2.1 Descripción de la Implementación

Se describen en este apartado los componentes de Gestión de Entrada/Salida y Gestión de Tiempos por ser los de mayor interés. El componente de Gestión de Concurrencia es una implementación típica y no se describe aquí; como componente de Gestión de Memoria se ha utilizado el proporcionado por la herramienta de desarrollo.

5.3.2.1.1 Gestión de Entrada/Salida

Los puntos de acceso del Subsistema de Pruebas con el Subsistema Inferior y con el Subsistema de Operación y Administración se han implementado en un único componente. La implementación se ha basado en la tecnología TCP/IP, haciendo uso de *sockets*. Esta elección facilita la portabilidad a otras plataformas, pues la mayoría de ellas soporta la interfaz *socket*. El Módulo Adaptador de las Pruebas funciona como servidor del Subsistema de Operación y Administración y como cliente del Subsistema Inferior.

La comunicación con el Subsistema de Operación y Administración se realiza a través de la entrada y salida estándares, lo que permite utilizar también una interfaz de consola en lugar de una interfaz gráfica.

En el caso de la comunicación con el Subsistema Inferior, para cada Punto de Control y Observación (PCO) definido en el Juego de Pruebas se crea un *socket*. Los *sockets* se almacenan en una lista que se puede gestionar a través de las siguientes funciones:

- `AdInicializaPCOs`: Inicializa las lista de PCOs.
- `AdNuevoPCO`: Añade un nuevo PCO a la lista y lo conecta con el otro extremo.
- `LocalizaPCO`: Busca un PCO en la lista.

El código que genera la herramienta *Tau* no configura todos los PCOs al iniciar la ejecución de un Caso de Prueba, sino que configura cada PCO cuando se realiza el

SPICUM servicio de publicaciones

primer envío por él. Se ha solventado este problema haciendo que la lista de PCOs se rellene durante la inicialización.

Cuando se solicita un envío, se llama a la función `GciSend`, dentro de la cual se realiza la codificación con la sintaxis de transferencia correspondiente (Sección 5.5.2.2.1). En recepción, la semántica de TTCN indica que se deben bloquear las colas de recepción, incluyendo la de posibles vencimientos de temporizadores, y comprobar las alternativas posibles para los datos que existan en las colas. A continuación, se desbloquean las colas y, si ninguna alternativa se ha cumplido, se repite el proceso. Esta semántica se implementa mediante la función `GciSnapshot`, que selecciona el primer evento y lo descodifica.

También es necesario considerar que las Operaciones del Juego de Pruebas requieren a veces comunicarse con el Subsistema Inferior. Para no modificar el conjunto de primitivas definido para cada PCO se emplea el PCO `INTERNO` (Capítulo 4, Sección 4.5.1.2.4), que está disponible para todos los Juegos de Pruebas. Adicionalmente, se han añadido comandos (`AVISO`, `RESPUESTA`, `SINO`) que permiten solicitar una acción por parte del operador y obtener su respuesta.

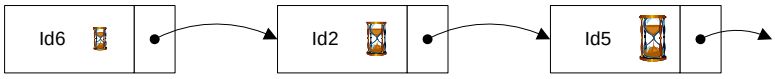
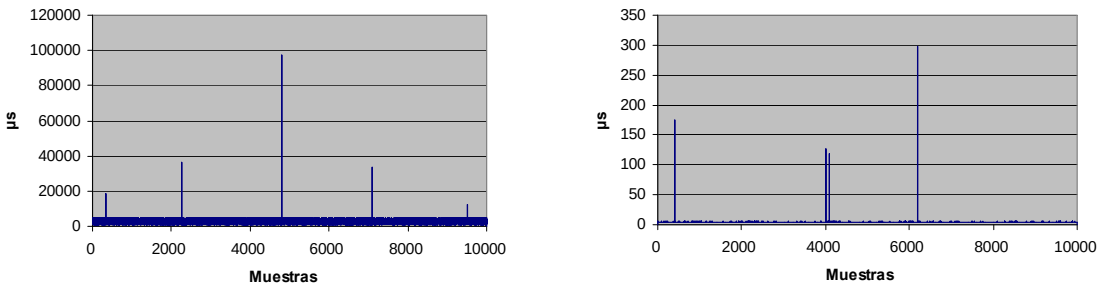


Figura 5.12: Representación del almacenamiento de la lista de temporizadores activos.

5.3.2.1.2 Gestión de tiempos

La Interfaz GCI define funciones para crear, activar, cancelar y leer un temporizador. La implementación que ofrece la herramienta guarda los temporizadores activos en una lista ordenada, siendo el primero de la lista el temporizador que vencerá antes (Figura 5.12). La gestión de esta lista no ha sido necesario modificarla, pero ha habido que adaptar las funciones de lectura del reloj de la plataforma para su uso en plataformas Windows.

La portabilidad ha obligado a modificar las funciones `AdInitTimers` y `AdGetElapsedTime` con un mecanismo que permita obtener la suficiente resolución. La función `ftime` ofrece resoluciones del orden de 55 ms en Windows 95 y Windows 98 y una resolución de 10 ms en Windows NT. Esta resolución puede ser suficiente para algunas aplicaciones, pero no es válida en general.



(a) *(b)*

Figura 5.13: Resolución del contador de alta resolución en (a) Pentium II a 200 MHz y (b) Pentium IV a 2400 MHz.

SPICUM servicio de publicaciones

La solución ha sido utilizar el contador de alta resolución hardware del sistema, mediante la funciones `QueryPerformanceFrequency` y `QueryPerformanceCounter` del API de Windows. Estas funciones permiten obtener resoluciones en torno a los 5 µs, aunque no existe una garantía. Estas funciones se han probado en equipos Pentium II y Pentium IV; la Figura 5.13 muestra las diferencias entre dos llamadas consecutivas a la función `QueryPerformanceCounter` en una batería de 10000 llamadas. Se puede observar que, aunque este contador no garantiza una resolución temporal mínima, sí ofrece unas prestaciones adecuadas para la gran mayoría de situaciones.

5.3.3 Módulo de Gestión de las Pruebas

El Módulo de Gestión de las Pruebas implementa la funcionalidad necesaria para el control de las Pruebas, el registro de trazas y la codificación de las interfaces externas del Subsistema de Pruebas. El Módulo de Gestión de las Pruebas (Figura 5.14) está dividido en tres componentes (Capítulo 3, Sección 3.2.2):Control de las Pruebas, Registro y Codec $Local_{TTCN}$.

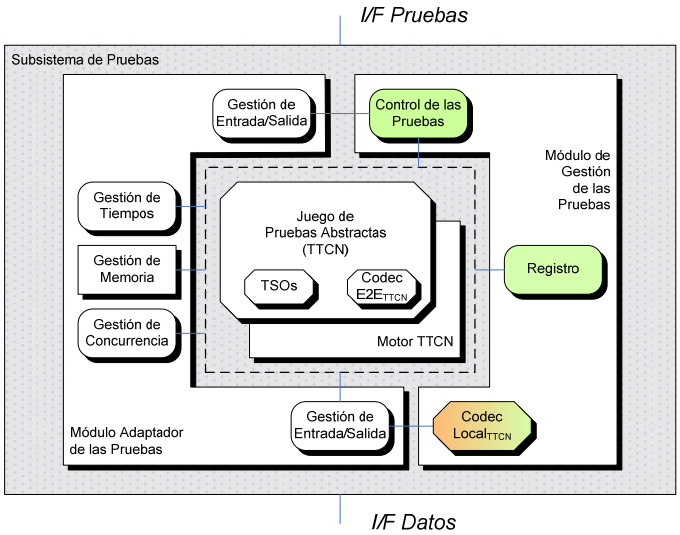


Figura 5.14: Componentes del Módulo de Gestión de las Pruebas.

La Interfaz de Pruebas permite el control de la selección y ejecución de las Pruebas. Los comandos definidos para esta interfaz se listan en la Tabla 5.4. Los dos últimos comandos `RESPUESTA` y `SINO` permiten solicitar del operador una acción durante la ejecución de las Pruebas.

5.3.3.1 Descripción de la Implementación

Se describen en este apartado los componentes Codec $Local_{TTCN}$ y Registro. No se incluye la descripción del componente Control de las Pruebas ya que es conceptualmente simple.

SPICUM servicio de publicaciones

Tabla 5.4: Comandos disponibles en la Interfaz de Pruebas y ejemplos de uso.

Comando	Descripción	Secuencia[8]
AVISO	Envía un mensaje al operador	< AVISO < Mensaje operador > OK
EJECUTA	Ejecuta el Caso de Prueba indicado, o los Casos de Prueba incluidos en el Grupo de Pruebas indicado	> EJECUTA Pruebas [Inicio de la ejecución] ... < LOG Mensaje ... [Ejecución finalizada] < VEREDICTO Vered
FIN	Finaliza la ejecución	> FIN < FIN
LISTA_CASOS	Solicita los Casos de Prueba disponibles	> LISTA_CASOS < Caso1 < Caso2 ... < .
LISTA_GRUPOS	Solicita los Grupos de Pruebas disponibles	> LISTA_GRUPOS < Grupo1 < Grupo2 ... < .
LISTA_PCOs	Solicita los PCOs del Juego de Pruebas	> LISTA_PCOs < Pco1 < Pco2 ... < .
PCOs	Configura los PCOs de las Pruebas	> PCOs > Pco1 param1 param2 < OK > Pco2 param1 param2 < OK > . < OK
PIXIT	Indica el archivo de Parámetros de Pruebas a utilizar	> PIXIT pixit.pic < OK
RESPUESTA	El operador elige entre una de varias opciones	< RESPUESTA < Opcion 1 < ... < Opcion n > Opcion x
SINO	Particularización del comando 'RESPUESTA' para una respuesta sí o no.	< SINO < Mensaje > SI_o_NO
¿?	Comando desconocido	> ¿? < DESCONOCIDO

5.3.3.1.1 Codec Local$_{TTCN}$

La herramienta proporciona dos sintaxis de transferencia para la comunicación entre el Juego de Pruebas y el Subsistema Inferior: una sintaxis de transferencia basada en una codificación ASCII y una sintaxis de transferencia basada en reglas de codificación estándar BER o PER. Ejemplos de ambas sintaxis de transferencia se encuentran en la sección 5.5.2.2.1. La sintaxis de transferencia ASCII proporcionada es muy básica y no cubre todos los tipos de datos utilizables, por lo que requiere un mayor esfuerzo de desarrollo. Además, produce codificaciones menos compactas y, sobre todo, no sigue

[8] '>' indica recepción en el Adaptador de Pruebas; '<' indica transmisión desde el Adaptador de Pruebas.

SPICUM servicio de publicaciones

ningún estándar internacional. Parte de estas limitaciones se solucionan con la librería proporcionada por la herramienta para la sintaxis de transferencia BER o PER; esta librería permite el uso de codificaciones alineadas o no y maneja todos los tipos ASN.1 y TTCN.

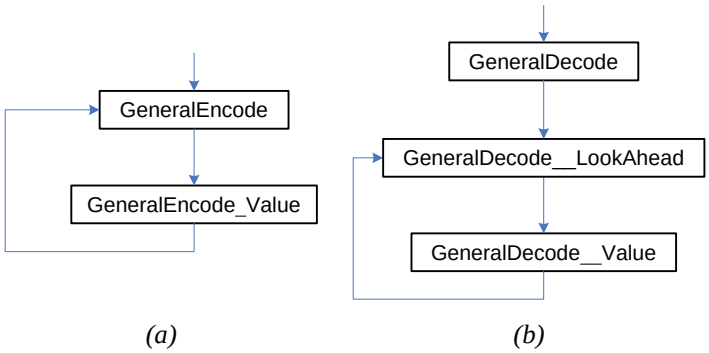


Figura 5.15: Proceso de (a) codificación y (b) descodificación para la sintaxis de transferencia ASCII.

Se han utilizado ambas sintaxis de transferencia ya que la segunda, aunque más potente, sólo ha estado disponible a partir del año 2000. La sintaxis de transferencia ASCII se ha utilizado en el Sistema de Pruebas DECT, mientras que la sintaxis de transferencia PER se ha empleado en el Sistema de Pruebas UMTS.

En el caso de la sintaxis de transferencia ASCII, el proceso funciona de la siguiente forma (Figura 5.15). Al enviar un valor, se invoca a la función `GeneralEncode`, la cual llama a la función `GeneralEncode_Value`. Dentro de esta función se determina el tipo del valor a codificar y se realiza la codificación; si es un tipo estructurado, se procesa cada uno de los campos o elementos secuencialmente. La codificación se realiza en una pasada. En recepción, la descodificación, sin embargo, utiliza una función adicional, `GeneralDecode__LookAhead`, ya que necesita decidir cuál es el tipo del valor que viene a continuación. Esta función mira cuál es el siguiente carácter no blanco y llama a la función `GeneralDecode__Value` con el identificador del tipo adecuado.

En el caso de la sintaxis de transferencia BER/PER, sólo hay que implementar las rutinas `GeneralEncode` y `GeneralDecode`, que invocan a las funciones `PER_ENCODE` y `PER_DECODE` que proporciona la librería. La codificación y descodificación se realizan mediante búferes que hay que inicializar y gestionar. La Figura 5.16 muestra un ejemplo del código resultante para la descodificación.

```
GciValue *GeneralDecode (unsigned char *segment)
{
  ...
  BaseBufInitBuf(&buf, bms_SmallBuffer);
  BufInitWriteMode( buf );
  BufPutSeg( buf, segment+OFFSET_HEADER, 4 );
  BufCloseWriteMode( buf );
  BufInitReadMode( buf );
  BufSetRule( buf, er_PER | er_Unaligned);
  PER_DECODE(buf, asp_types[asp_id].ASN1TypeInfo,
     UCF_PER_DEFAULT_ENCODING_VARIANT, &length_aux, &asp);
  BufCloseReadMode( buf );
  ...
}
```

Figura 5.16: Código ejemplo de la descodificación para la sintaxis de transferencia BER/PER.

SPICUM servicio de publicaciones

La herramienta *GenCod* (Sección 5.5.2.2) genera una tabla (`asp_types`) con todas las primitivas utilizadas y, para cada una de ellas, la referencia al correspondiente tipo de datos.

5.3.3.1.2 Registro

La herramienta utiliza los eventos de registro definidos en la Interfaz GCI, aunque con ligeras modificaciones (Tabla 5.5). El generador de código se encarga de insertar las correspondientes líneas en los ficheros de salida para que estos eventos se notifiquen durante la ejecución de una Prueba. Estas trazas invocan a la función `GciLog`, en la cual es posible decidir el tratamiento otorgado a cada mensaje. Se permiten dos opciones, bien que las trazas se envíen al Subsistema de Operación y Administración a través de la salida estándar, de forma que se puedan mostrar en tiempo de ejecución, o bien guardarlas en un archivo para su posterior análisis.

Tabla 5.5: Conjunto de eventos que se notifican durante la ejecución de un Caso de Prueba.

Identificador	Descripción	Identificador	Descripción
`GciLogActivate`	Activar rama por defecto	`GciLogReadT`	Leer temporizador
`GciLogAssign`	Asignación	`GciLogRec`	Datos recibidos[9]
`GciLogCancelT`	Cancelar temporizador	`GciLogRecE`	Evento de recepción de datos[9]
`GciLogCreate`	Crear Componente de Prueba	`GciLogSendE`	Evento de envío
`GciLogDone`	Detener Componente de Prueba	`GciLogStartDEF`	Iniciar rama por defecto
`GciLogEnterAttach`	Iniciar rama	`GciLogStartT`	Iniciar temporizador
`GciLogError`	Error	`GciLogStartTC`	Iniciar Caso de Prueba
`GciLogExitAttach`	Detener rama	`GciLogStartTS`	Iniciar Paso de Prueba
`GciLogGoto`	Goto	`GciLogStopDEF`	Detener rama por defecto
`GciLogImplSend`	Envío implícito	`GciLogStopT`	Detener temporizador
`GciLogMatch`	Comparación exitosa	`GciLogStopTC`	Detener Caso de Prueba
`GciLogMatchFailed`	La comparación falló[10]	`GciLogStopTS`	Detener Paso de Prueba
`GciLogMessage`	Mensaje general	`GciLogTimeout`	Vencimiento de temporizador[9]
`GciLogNoMatch`	La comparación con una línea falla	`GciLogTimeoutE`	Evento de vencimiento de temporizador[9]
`GciLogOtherE`	Comparación con cualquier cosa	`GciLogVerdict`	Veredicto Final
`GciLogPVerdict`	Veredicto parcial		

5.4 Componentes del Subsistema Inferior

Como se ha visto en la descripción de la Arquitectura, el Módulo de Protocolos del Subsistema Inferior requiere para su ejecución de un Módulo Adaptador de los

[9] Los identificadores terminados en 'E' se notifican cuando la línea de evento TTCN se ejecuta; los otros identificadores se usan cuando la acción sucede a bajo nivel, a través de la función correspondiente de la interfaz GCI. En una ejecución normal los identificadores terminados en 'E' deberían notificarse después de los otros, posiblemente con otros mensajes intercalados.

[10] Este evento es diferente del evento `NoMatch` porque significa que los datos recibidos no concuerdan con ninguna de las posibles alternativas, por lo que se debe asignar un veredicto diferente de `PASS`, y el Caso de Prueba finalizará.

SPICUM servicio de publicaciones

Protocolos y un Módulo de Gestión de los Protocolos. Estos Módulos han sido diseñados para que el Módulo de Protocolos se pueda ejecutar en las plataformas Linux, Unix y Windows[11]. También se han implementado estos módulos para la plataforma DSP/BIOS (Capítulo 8, Sección 8.6.2).

5.4.1 Módulo Adaptador de los Protocolos

El Módulo Adaptador de los Protocolos implementa la funcionalidad necesaria para que el Subsistema Inferior se pueda ejecutar sobre una determinada plataforma. Se trata de funciones generales, normalmente de bajo nivel, como temporizadores, gestión de colas, gestión de memoria, etc. El Módulo Adaptador de los Protocolos (Figura 5.17) está dividido en cuatro componentes (Capítulo 3, Sección 3.2.3): Gestión de Entrada/Salida, Gestión de Tiempos, Gestión de Memoria y Gestión de Concurrencia. Los tres últimos componentes son proporcionados por la herramienta de desarrollo. El componente Gestión de Entrada/Salida ha sido desarrollado conforme a las necesidades del diseño.

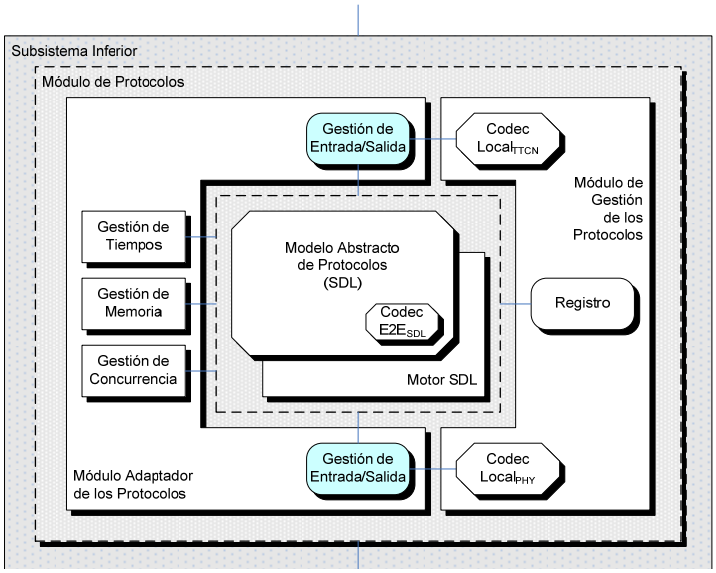


Figura 5.17: Componentes del Módulo Adaptador de los Protocolos.

5.4.1.1 Gestión de Entrada/Salida

Este componente realiza la comunicación del Módulo de Protocolos con el Subsistema de Pruebas y el Módulo de Capa Física. Se ha diseñado de forma que sea un componente genérico; por ello, se ha separado la funcionalidad de control de las vías de entrada y salida de la funcionalidad de codificación. Así, este componente se limita a enviar y recibir datos, invocando a las funciones de codificación y descodificación, respectivamente.

Las principales funciones (Figura 5.18) son las siguientes:

- `xInitEnv`: Inicializa las conexiones de comunicación con el exterior. Los puertos a utilizar, y los nombres de los canales correspondientes, se leen de un fichero de configuración, tras lo cual se crean las conexiones. En la

[11] Durante la fase de enlazado, para las dos primeras plataformas hay que incluir las librerías `libsocket` y `libnsl`; para la tercera plataforma, hay que definir la opción de compilación `PLATAFORMA_WIN32` e incluir la librería `libwsock32`.

SPICUM servicio de publicaciones

comunicación con el Subsistema de Pruebas actúa como servidor. Si es necesario, inicializa también la conexión con el Subsistema Inferior.

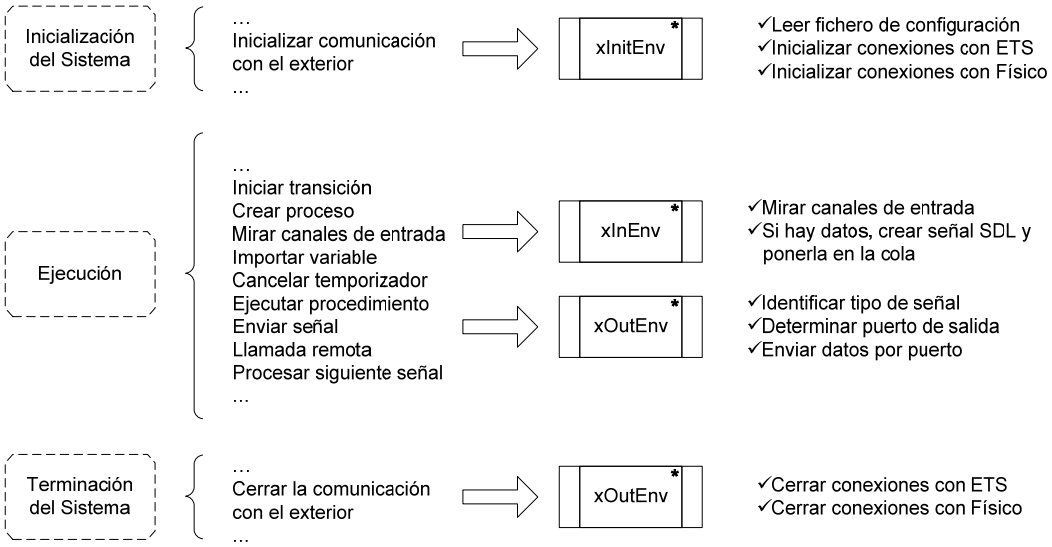


Figura 5.18: Funciones del componente Gestión de Entrada/Salida.

Las conexiones con el Subsistema de Pruebas se realizan a través de *sockets*, usando la API habitual de las funciones `socket`, `bind`, `connect`, `listen` y `accept`. Los *sockets* se almacenan en una estructura junto con su nombre para posterior utilización. En Windows el uso de esta API requiere una inicialización adicional, con el código siguiente:

```
wVersionRequested = MAKEWORD( 2, 0 );
err = WSAStartup( wVersionRequested, &wsaData );
```

- `xCloseEnv`: Cierra las conexiones de acceso al sistema y libera los recursos.

```
if ( select(MAXDESCRIPTOR, &readfds, 0, 0, &t) > 0 ) {
  for(i=0; i<numeroCanales; i++) {
    if (FD_ISSET(obtenerCanalPorID(i)->socket, &readfds))
    {
      ...
      val = recv(obtenerCanalPorID(i)->socket, &buf, 1, 0);
      ...
      #ifdef SINTAXIS_ASCII
      error = Decod_Primitivas (buf);
      #else  -- SINTAXIS_BER_PER
      DecodeASP(buf, longitud, tBuf);
      #endif
      ...
    }
  }
}
```

Figura 5.19: Esquema del procesamiento en la función `xInEnv`.

- `xInEnv`: Comprueba todos los posibles puertos de entrada del sistema, para determinar si se ha recibido algo; si es así, construye una señal SDL y la encola en el canal correspondiente. Es llamada por el planificador de SDL cuando no se

SPICUM servicio de publicaciones

está ejecutando ninguna transición. La Figura 5.19 muestra un esquema del código para determinar y recibir datos del Juego de Pruebas. Tras la recepción, con `recv`, se procede a la descodificación de los datos recibidos. Las funciones para la descodificación[12] son distintas según se utilice la sintaxis de transferencia ASCII (función `Decod_Primitivas`) o una sintaxis de transferencia BER/PER (función `DecodeASP`).

- `xOutEnv`: Esta función es llamada cuando se desea enviar una señal hacia el exterior del sistema SDL. Primero se codifica la información de la señal[12], y luego se envía por el puerto correspondiente; un esquema del algoritmo se muestra en la Figura 5.20.

```
...
#ifdef SINTAXIS_ASCII
Codif_Primitivas (S,OutStr);
#else  -- SINTAXIS_BER_PER
EncodeASP(S, tBuf, data, &longitud);
#endif
...
send(obtenerCanal(canal)->socket, OutStr, longitud, 0);
```

Figura 5.20: Esquema del procesamiento en la función `xOutEnv`.

5.4.2 Módulo de Gestión de los Protocolos

El Módulo de Gestión de los Protocolos implementa la funcionalidad necesaria para el registro de trazas y la codificación de las interfaces externas del Subsistema de Inferior. Está dividido en tres componentes (Figura 5.14) (Capítulo 3, Sección 3.2.3): Codec $Local_{TTCN}$, Registro y Codec $Local_{PHY}$.

Se describe a continuación el componente Registro. El Codec $Local_{TTCN}$ se describe en la Sección 5.5.2; el Codec $Local_{PHY}$ es particular de cada Sistema de Pruebas.

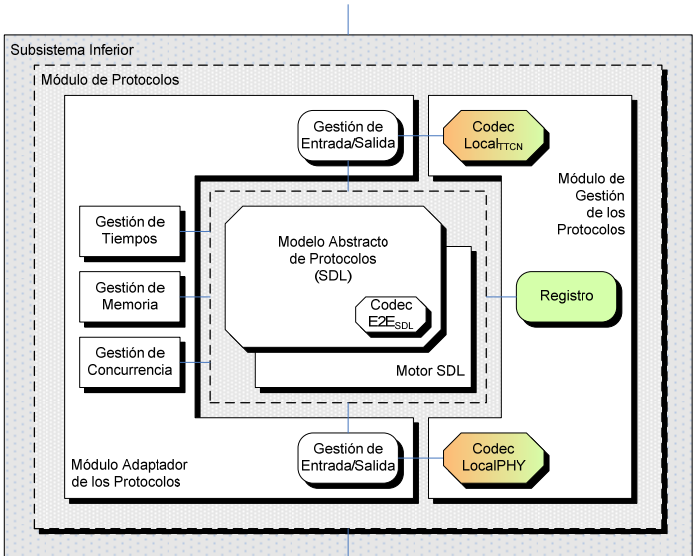


Figura 5.21: Componentes del Módulo de Gestión de los Protocolos.

[12] Las funciones de codificación y descodificación deben ser generadas para cada Sistema de Pruebas. En el módulo de Gestión de Entrada/Salida sólo se incluyen las llamadas a estas funciones.

SPICUM servicio de publicaciones

5.4.2.1 Registro

El componente Registro permite obtener trazas tanto de operación como de depuración de la ejecución del Módulo de Protocolos. Los eventos que se pueden trazar son los indicados en la Tabla 5.6. Estas trazas se han implementado dentro de las funciones `xSignalLog` y `xProcessLog`, que son invocadas por el planificador del sistema (`xMainLoop`), al seleccionar la siguiente transición a ejecutar (`SDL_Execute`) y al enviar una señal (`SDL_Output`).

Tabla 5.6: Eventos que se registran en el Módulo de Gestión de los Protocolos.

Elemento	Evento	Información
Señal	Envío	Nombre, parámetros, origen, destino, canal
	Recepción	
Bloque	Envío de señal	Destino
	Recepción de señal	Origen
Proceso	Creación	---
	Destrucción	---
	Cambio de estado	Nuevo estado
	Ejecución de tarea	Símbolo y código de la tarea ejecutada

5.5 Generadores Automáticos

La comunicación entre dos entidades de una red o sistema implica la definición de una semántica común para la información, que se define de forma abstracta de manera que cada parte puede representar la información a su propio estilo, una sintaxis abstracta. A la hora de comunicarse esta semántica debe ser representada en un formato que ambas entidades comprendan, una sintaxis de transferencia. La traducción entre la representación interna y la representación común requiere del uso de un codec (elemento codificador/descodificador). La implementación de una interfaz consta, pues, de dos pasos:

a) Definición de la interfaz y

b) Construcción de las rutinas de codificación y descodificación.

Cuando se trata de pocos mensajes es viable una implementación manual pero, al aumentar su número, esta vía es tediosa y, sobre todo, propensa a errores. La implementación automática permite soslayar estos inconvenientes; además, en caso de modificación de la interfaz, la implementación automática permite regenerar el codec rápidamente.

Los Juegos de Pruebas, vistos como sustitutos de uno o más niveles de un sistema de comunicaciones, emplean dos tipos de interfaces. Una interfaz local con los niveles adyacentes, interna del Sistema de Pruebas, y una interfaz extremo-extremo constituida por los mensajes intercambiados entre el Caso de Prueba y la Implementación Bajo Prueba. Se ha trabajado en la automatización de ambos tipos de interfaces, para lo que se han diseñado un Generador de Interfaces Locales (*GenInt*) y un Generador de Codificador/Descodificador para la Interfaz Aire (*GenCodecAir*).

En los siguientes apartados se describen los generadores automáticos de interfaces que se han implementado. En primer lugar, se presenta una visión general. A continuación, se describe el Generador de Interfaces Locales (*GenInt*), constituido por dos

SPICUM servicio de publicaciones

herramientas que cubren los dos aspectos anteriormente mencionados en la implementación de una interfaz entre Subsistemas locales. Tras ello, se expone la herramienta *GenCodecAir* construida para generar automáticamente los procedimientos de codificación y descodificación de mensajes extremo a extremo.

5.5.1 Visión Global

Esta sección es un resumen del contenido de las secciones 5.5.2 y 5.5.3 para aquellos lectores que no deseen profundizar en la implementación de los generadores automáticos.

El Generador de Interfaces Locales (*GenInt*) permite implementar automáticamente la interfaz entre el Subsistema de Pruebas y el Subsistema Inferior. Está integrado por dos herramientas: el Generador de la Definición de la interfaz (*GenDef*), que extrae el conjunto de tipos de datos empleados en la interfaz del Subsistema Inferior a partir de la interfaz del Subsistema de Pruebas, y el Generador de Codificador de la interfaz (*GenCod*), que automatiza la construcción de los correspondientes codificadores y descodificadores en ambos componentes.

Estas herramientas han evolucionado al ritmo que lo han hecho los Juegos de Pruebas y los entornos de desarrollo comerciales. Por ello, inicialmente hacían uso de una sintaxis de transferencia ASCII y sólo aceptaban tipos de datos TTCN; su implementación final utiliza una sintaxis de transferencia BER/PER.

La herramienta *GenDef* procesa los ficheros en formato `MP` de los Juegos de Pruebas y genera un conjunto de ficheros que contienen las definiciones de las primitivas (ASPs), unidades de datos (PDUs) y tipo de datos utilizados en la interfaz entre el Subsistema de Pruebas y el Subsistema Inferior. Para construir esta herramienta se ha utilizado el generador de analizadores *PRECCX* [BREU97].

La salida generada por la herramienta *GenDef* es sólo uno de los dos pasos necesarios para lograr comunicar el Subsistema de Pruebas con el Subsistema Inferior. La información que circula entre ambos Subsistemas debe ser codificada como paso previo a su transmisión; en recepción, se debe proceder a su descodificación. La herramienta *GenCod* permite generar, de forma automática, el codificador que existe en el extremo del Subsistema Inferior.

El Generador de Codificador de la interfaz (*GenCod*) se ha construido como una herramienta autónoma, ya que en algunas ocasiones es interesante poder modificar la información extraída por *GenDef*. Las entradas al generador se indican en un fichero de texto donde se listan los archivos a emplear; estos archivos son los ficheros de tipos creados por *GenDef*.

La codificación y descodificación se realizan, respectivamente, en las funciones que se encargan de la transmisión (`xOutEnv`) y de la recepción (`xInEnv`) de las señales que constituyen la comunicación externa del sistema SDL (ver sección 5.4.1.1). El Módulo Adaptador de los Protocolos está implementado en C, lenguaje que permite una más cómoda manipulación de la información, tanto a la hora de codificarla como de descodificarla, que el lenguaje SDL.

La tercera herramienta diseñada es el Generador de Codificador/Descodificador para la Interfaz Aire (*GenCodecAir*), que genera automáticamente los procedimientos de codificación y descodificación de los mensajes extremo-extremo de los Niveles de Red y de Enlace. Estos procedimientos se han implementado en SDL, ya que han sido pensados para su integración en el modelo del Módulo de Protocolos. La herramienta

SPICUM servicio de publicaciones

está adaptada a la sintaxis de transferencia del Sistema DECT ([ETS 300 175-4], [ETS 300 175-5]).

Como entradas a este generador se emplean los archivos que contienen las definiciones de los tipos de datos creados previamente por la herramienta *GenDef*: ficheros `*.asp`, `*.pdu` y `*.tad` (ver sección 0). Como resultado se obtiene un fichero en formato textual (`PR`), que puede ser utilizado en el modelo SDL. Si la herramienta lo permite, se puede convertir este archivo a formato gráfico (`GR`) para un manejo más cómodo[13].

5.5.2 Generador de Interfaces Locales

La integración de los distintos componentes en un Sistema de Pruebas requiere de la comunicación entre el Subsistema de Pruebas y el Subsistema Inferior. La Figura 5.22 muestra ejemplos de la sintaxis abstracta en cada Subsistema y de posibles sintaxis de transferencia. Se han construidos dos herramientas para procesar esta interfaz. La herramienta *GenDef* (Generador de la Definición de la interfaz) permite generar el conjunto de tipos de datos empleados en la interfaz del Subsistema Inferior a partir de la interfaz del Subsistema de Pruebas. La herramienta *GenCod* (Generador de Codificador de la interfaz) automatiza la construcción de los correspondientes codificadores y descodificadores en ambos componentes, adaptados a un determinado Subsistema de Pruebas. El conjunto de estas herramientas se ha denominado *GenInt* (Generador de Interfaces).

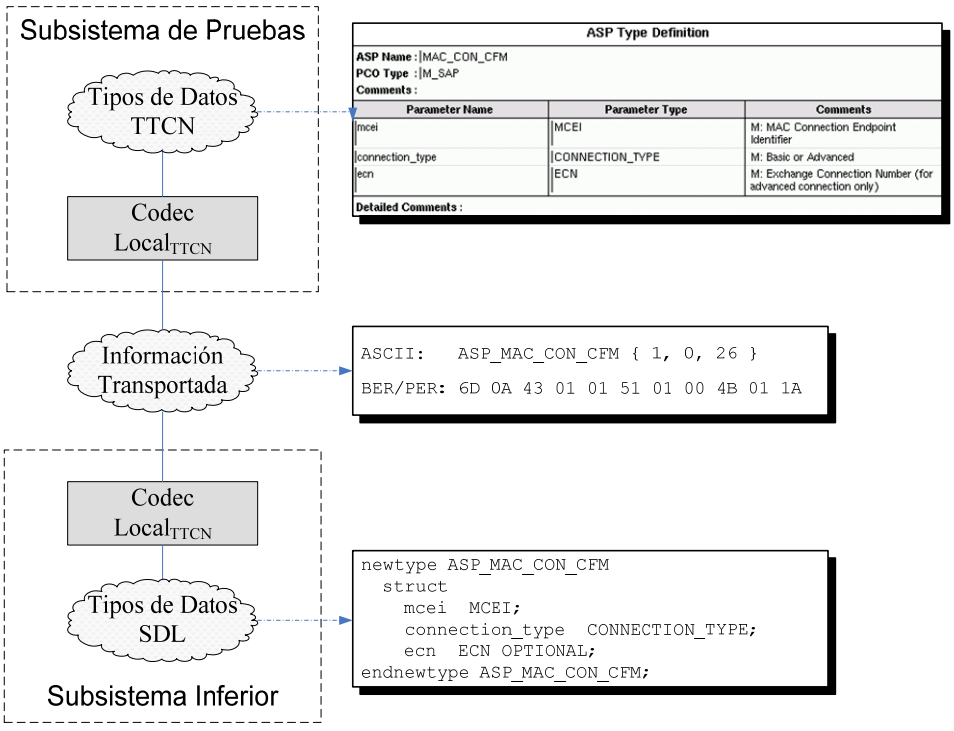


Figura 5.22: Esquema de la interfaz entre el Subsistema de Pruebas y el Subsistema Inferior.

Estas herramientas han evolucionado al ritmo que lo han hecho los Juegos de Pruebas y los entornos de desarrollo comerciales. Por ello, inicialmente hacían uso de una sintaxis de transferencia ASCII y sólo aceptaban tipos de datos TTCN; su implementación final

[13] De ahí el aspecto en formato gráfico, GR, del código SDL del codificador, pues se genera tras expandir las macros.

SPICUM servicio de publicaciones

utiliza una sintaxis de transferencia BER/PER. La Figura 5.23 muestra el flujo de información durante el uso de la herramienta *GenInt*.

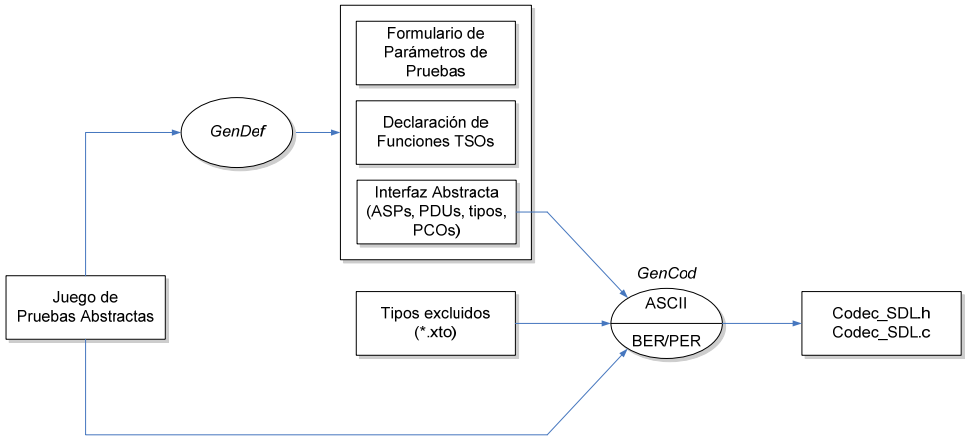


Figura 5.23: Esquema de utilización del Generador de Interfaces.

5.5.2.1 Generador de la Definición de la Interfaz

Como se ha comentado en el (Capítulo 4, Sección 4.5.1.2.1), uno de los aspectos más problemáticos es la definición de la interfaz superior del Subsistema Inferior, ya que esta interfaz viene fijada en los Juegos de Pruebas. Es, por tanto, conveniente utilizar esta interfaz desde las primeras fases del desarrollo del Subsistema Inferior. Para ello, *GenDef* es capaz de extraer la interfaz que ofrece un Juego de Pruebas.

La interfaz viene definida por los siguientes elementos:

- Primitivas Abstractas de Servicio (ASPs)
- Unidades de Datos de Protocolo (PDUs)
- Puntos de Control y Observación (PCOs)
- Operaciones del Juego de Pruebas (TSOs)
- Parámetros de Pruebas
- Tipos de Datos

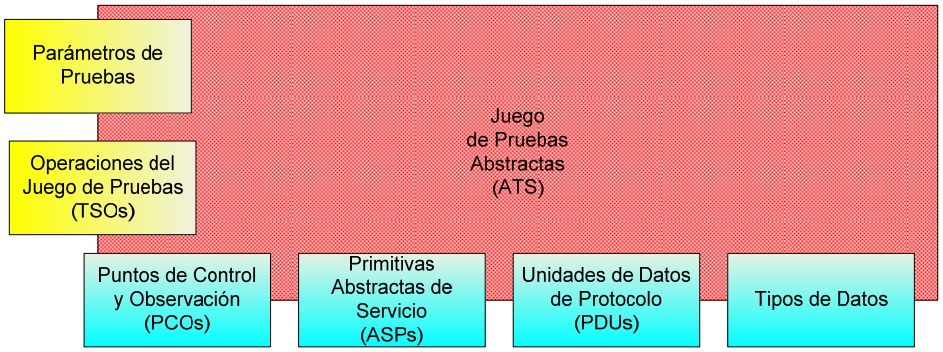


Figura 5.24: Elementos de la interfaz de un Juego de Pruebas Abstractas.

Al Subsistema Inferior le son de interés la definición de las primitivas, las unidades de datos, los Puntos de Control y Observación y los tipos de datos utilizados en sus

SPICUM servicio de publicaciones

definiciones. La herramienta, adicionalmente, extrae la definición de las funciones TSO, que deben ser implementadas, y la definición de los Parámetros de Pruebas.

5.5.2.1.1 Definición de la gramática de TTCN

La sintaxis de un lenguaje, natural o de programación, se puede definir mediante la descripción del conjunto de reglas gramaticales permitidas en el lenguaje. A las notaciones que permiten describir estas reglas se las denomina metalenguajes o metanotaciones.

Ejemplo de reglas BNF

```
S  := '-' FN |
      FN
FN := DL |
      DL '.' DL
DL := D |
      D DL
D  := '0' | '1' | '2' | '3' | '4' | '5' | '6' | '7' | '8' | '9'
```

Ejemplo de reglas EBNF

```
S := ['-'] D+ [('.' D+)]
D := '0' | '1' | '2' | '3' | '4' | '5' | '6' | '7' | '8' | '9'
```

Figura 5.25: Ejemplo de reglas equivalentes en notación BNF y EBNF [GARS01] para describir números reales.

En el caso de TTCN, su gramática viene formalmente descrita en notación EBNF (*Extended Backus-Naur Form*), extensión de la notación BNF[14] (*Backus-Naur Form*). Estos metalenguajes ofrecen un mecanismo para describir formal y matemáticamente una gramática. Ambas notaciones tienen la misma potencia, lo que quiere decir que cualquier regla EBNF se puede transformar en un conjunto equivalente de reglas BNF. Sin embargo, EBNF resulta en unas reglas gramaticales más concisas y fáciles de leer.

La descripción de las diferencias entre ambas notaciones queda fuera de este ámbito, pero sirva como ejemplo la Figura 5.25 que muestra la descripción sintáctica de un número real en ambas notaciones. Una descripción completa de la notación EBNF estandarizada se puede encontrar en [ISO 14977], aunque la gramática de TTCN utiliza una versión ligeramente modificada de esta notación. La gramática de TTCN, por el hecho de poder expresarse en notación BNF, es independiente del contexto.

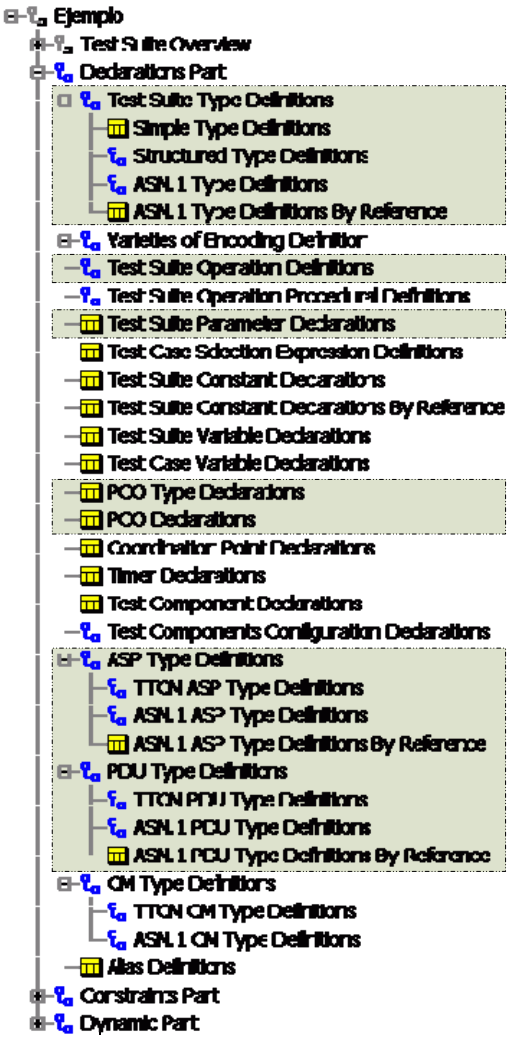
5.5.2.1.2 Declaración de la interfaz en un Juego de Pruebas

Un Juego de Pruebas tiene la estructura mostrada en la Figura 5.26-a; en esta figura se han resaltado las secciones que incluyen declaraciones relacionadas con la interfaz del Juego de Pruebas. En forma textual (formato `MP` – *Machine Processable*), cada rama tiene la estructura genérica que aparece en la Figura 5.26-b. Todas las declaraciones de un determinado tipo, como por ejemplo las primitivas, se encuentran recogidas correlativamente dentro de la misma sección. La herramienta *GenDef* procesa el fichero `MP` buscando los inicios de sección de cada uno de los grupos de elementos listados anteriormente y extrae la información. Las definiciones de estos grupos de elementos se

[14] La notación BNF fue desarrollada por John Backus y Peter Naur. Niklaus Wirth creó las extensiones de la notación EBNF [TUTEB].

SPICUM servicio de publicaciones

encuentran ubicados en la sección de Declaraciones y tienen el formato mostrado de la Figura 5.27 a la Figura 5.29.




```
<Palabra_Clave_de_Inicio_de_Sección>
<Sección_anidada>
       <Sección_anidada>
              ...
       <Sección_anidada>
              ...
       ...
<Sección_anidada>
       <Sección_anidada>
              ...
       <Sección_anidada>
              ...
       ...
...
<Sección_anidada>
       <Sección_anidada>
              ...
       <Sección_anidada>
              ...
       ...
<Palabra_Clave_de_Fin_de_Sección>
```

(a) *(b)*

Figura 5.26: (a) Estructura de la parte de Declaraciones de un Juego de Pruebas; (b) Estructura genérica de las secciones de un Juego de Pruebas en notación textual.

SPICUM servicio de publicaciones

Figura 5.27: Forma textual de las secciones de definición (a) de las Primitivas Abstractas de Servicio y (b) de las Unidades de Datos de Protocolo.

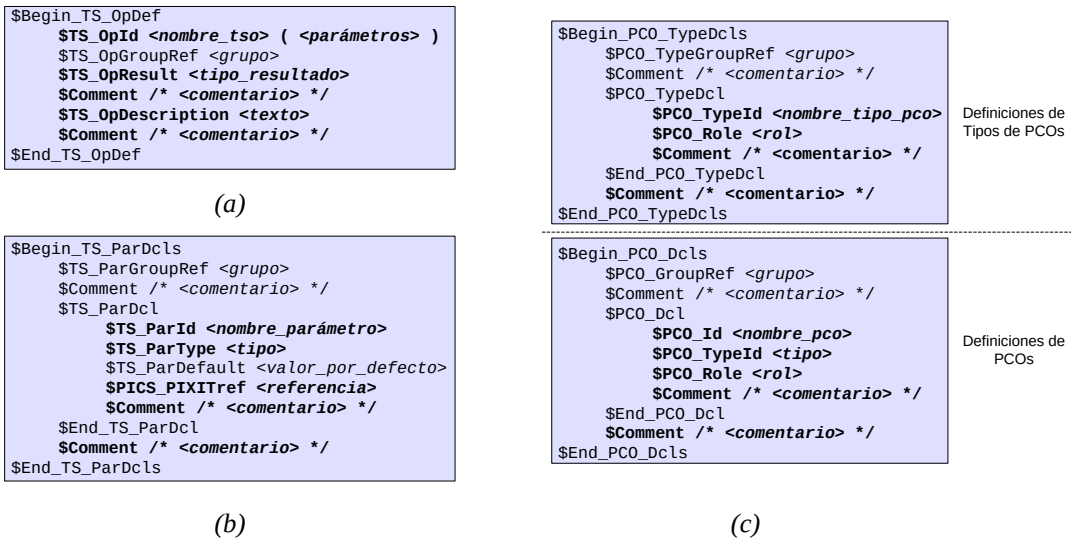


Figura 5.28: Forma textual de las secciones de definición de (a) las Operaciones del Juego de Pruebas, (b) los Parámetros de Pruebas y (c) los Puntos de Control y Observación.

SPICUM servicio de publicaciones

Figura 5.29: Forma textual de las secciones de definición de los Tipos de Datos.

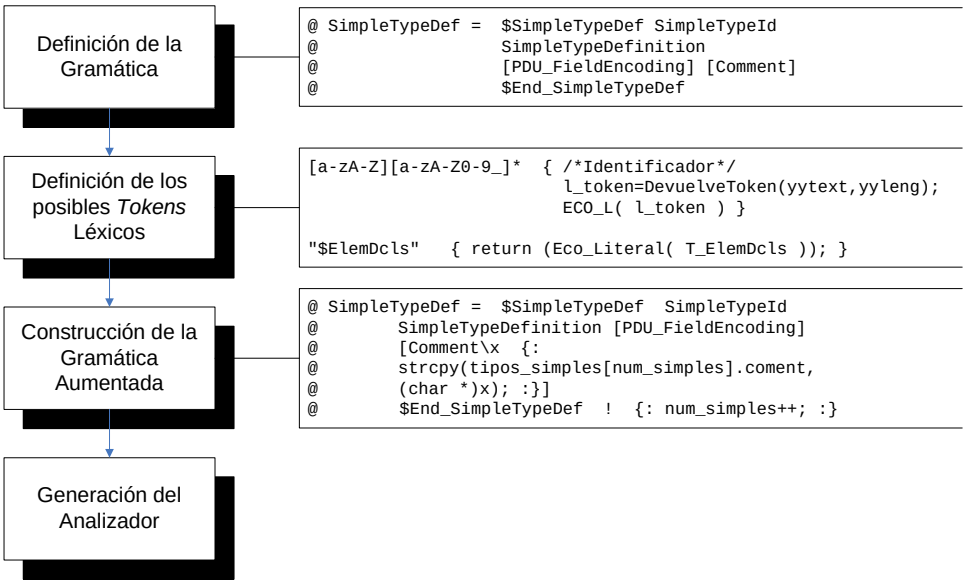


Figura 5.30: Proceso de generación de un analizador sintáctico.

5.5.2.1.3 Descripción de la implementación

La herramienta *GenDef* se ha construido haciendo uso del generador de analizadores *PRECCX* [BREU97]. A continuación se describe en detalle la herramienta construida y los resultados obtenidos; los pasos que se han dado se resumen en la Figura 5.30. El

SPICUM servicio de publicaciones

Apéndice E comenta posibles alternativas de implementación y describe brevemente el uso de *PRECCX*. La herramienta *GenDef* ha servido como base del trabajo [LINA01].

5.5.2.1.3.1 Construcción de la gramática y tokens léxicos

La implementación requiere como primer paso construir la gramática de TTCN, que se encuentra en la norma [X.292], anexo A. Esta gramática ha sido procesada para adaptarla al generador *PRECCX*. Aunque *PRECCX* permite la integración de un analizador léxico como *Lex* ([LESK75], [LEVI92]) o *Flex* [PAXS95], se ha utilizado el analizador léxico que viene ya incluido.

Las reglas de la gramática han sido adaptadas a la sintaxis de entrada esperada por *PRECCX*. Para ello, se ha construido el preprocesador *Gengramy*, que realiza las siguientes modificaciones:

- Semántica del carácter '$': En la gramática las palabras clave son de la forma `$Palabra_Clave`, pero en *PRECCX* este carácter indica el final de línea. Se ha creado una nueva regla para describir cada palabra clave como un *token*, donde se deletrea la palabra clave. Por ejemplo, para la palabra clave `$PDU_FieldDcls` se ha creado una nueva regla `_PDU_FIELDDCLS_`.
- Se ha modificado el símbolo que delimita el nombre de la regla de su descripción de '`::=`' a '`=`'.
- La numeración de las reglas se ha mantenido incluyéndola en comentarios.

```
@ palclave  = <'$'>\x {: pos = 0; texto[pos++] = x; :}
@             { { (isalpha) | <'_'> | (isdigit) }\c
@                 {: texto[pos++] = c; :} }*
@             {: texto[pos++] = '\0'; :}  {@ (char *)texto @}

@ numero = ^ (isdigit)\x {: pos = 0; texto[pos++] = x; :}
@           { (isdigit)\c {: texto[pos++] = c; :} }*   <' '>*
@           {: texto[pos++] = '\0'; :}  {@ (char *)texto @}

@ _PUNPUNEQ =  <':'> <':'> <'='>

@ expr =
@    palclave\x  {: otra_palclave((char *)x); :}
@         [ expr ]
@       | _PUNPUNEQ   {: fprintf(fiyacc, "="); :}
@         [ expr ]
@       | numero\x
@          {: fprintf(fiyacc, "\n\n/* %s */\n@ ", (char *)x); :}
@         [ expr ]
@       | >'\n'<\x    {: fprintf(fiyacc, "%c", (char)x); :}
@         [ expr ]

@ top  =
@       expr
@       | $             {: fprintf(fiyacc, "\n"); :}
@       | $$  !         {: printf("SALIENDO... Fin de fichero\n"); :}
```

Figura 5.31: Gramática del preprocesador Gengramy de la gramática TTCN disponible en [X.292].

SPICUM servicio de publicaciones

La gramática del preprocesador se muestra en la Figura 5.31, que sirve de entrada a PRECCX para generar el código C del preprocesador. El preprocesador se puede invocar con el comando

```
gengramy <gram.txt> [<gramtok.y>] [<gramreg.y>]
```

indicando el fichero que contiene la gramática y dos ficheros de salida, con sus nombres por defecto. El primer fichero (‘*gramreg.y*’) contiene las reglas de producción propiamente de la gramática y el segundo (‘*gramtok.y*’) las reglas que definen los *tokens* de las palabras claves . A estos ficheros hay que añadirle uno más (‘*gramlit.*y’) que incluye la definición de los literales no terminales presentes en la gramática como *tokens*, por ejemplo, `BOOLEAN`, `INT_TO_BIT`, `READ_TIMER`, etc.

```
/* Regla 396 - Tipo y atributos del tipo de un campo de una PDU */
@ TypeAndAttributes =
@         {Type\x {: strcpy(aux_tipoyattr.tipobase, (char *)x);
@         aux_tipoyattr.restric.tipo = RESTRIC_VACIO; :} bl
@         [LengthAttribute\y
@                {: aux_tipoyattr.restric = *((Restric *)y); :} ] }
@         {@ (char *)&aux_tipoyattr @}
@       | __PDU__  {: strcpy(aux_tipoyattr.tipobase, "");
@         aux_tipoyattr.restric.tipo = RESTRIC_PDU;
@         aux_tipoyattr.restric.dato.pdu = 1; :}
@         {@ (char *)&aux_tipoyattr @}

/* Regla 400 - Restricción de longitud de un tipo de datos */
@ RangeLength = <'['> bl LowerBound\x
@                {: if (*((int *)x) != -2) {
@                     aux_restric.dato.long_rango.min = *((int *)x);
@                     aux_restric.tipo = RESTRIC_LONG_RANGO; }
@                   else  {
@                     strcpy(aux_restric.dato.cadena, auxcad);
@                     aux_restric.tipo = RESTRIC_CADENA; }
@                :}
@               bl To bl UpperBound\y
@                {: if (*((int *)y) != -2) {
@                     aux_restric.dato.long_rango.min = *((int *)y);
@                     aux_restric.tipo = RESTRIC_LONG_RANGO; }
@                   else  {
@                     strcpy(aux_restric.dato.cadena, auxcad);
@                     aux_restric.tipo = RESTRIC_CADENA; }
@                :}
@                bl <']'>
@                {@ (char *)&aux_restric @}
```

Figura 5.32: Ejemplos de reglas de la gramática aumentada.

Algunas reglas han sido modificadas para adaptarse a las características propias del generador *PRECCX* como, por ejemplo, que la secuencia más larga debe preceder a las demás (regla 680), la recursión por la derecha es mejor que por la izquierda (regla 707), etc. También se han encontrado reglas a las cuales les faltaban campos, como, por ejemplo, las reglas 619 (`TestGroup`) y la 636 (`TestStep`). Además, el analizador léxico proporcionado por la propia *PRECCX* lee carácter a carácter y devuelve cada uno de ellos como un *token*. Por este motivo, no es posible definir el concepto `espacio_en_blanco` (uno o más espacios y/o final de línea) en el analizador léxico. Para tener esto en cuenta, se han modificado las reglas gramaticales incluyendo símbolos adicionales que modelan este concepto.

SPICUM servicio de publicaciones

5.5.2.1.3.2 Gramática aumentada

La gramática aumentada se construye incluyendo las acciones deseadas en las reglas del fichero '*gramreg.y*' obtenido anteriormente. Estas acciones se codifican dentro de la construcción '`{:...:}`' y pueden incluir llamadas a funciones C implementadas por el programador, por ejemplo, para almacenar el símbolo en una tabla. Una vez construida la gramática aumentada el aspecto de las reglas de producción es el mostrado en los ejemplos de la Figura 5.32. La construcción '`{@...@}`' permite que la regla devuelva un cierto valor como resultado de su procesamiento.

Antes de poder generar el analizador es necesario definir las estructuras de datos que almacenarán la información de interés y las funciones que gestionarán el procesamiento de estas estructuras. Las estructuras utilizadas son tablas que almacenan la información relevante de cada tipo de datos; un ejemplo de estas tablas se muestra en la Figura 5.33, que se utiliza para almacenar las primitivas de servicio definidas en un archivo TTCN.

```
typedef struct
{
  char nombre[TAM_NOMBRES];                /* Nombre */
  char coment1[TAM_COMENT+3];              /* Comentario 1 */
  char pco[TAM_NOMBRE_PCO];                /* PCO usado */
  int num_params;                          /* Número de parámetros */
  TipoSimple params[NUM_PARAMETROS_ASP];   /* Parámetros de la ASP */
  char coment2[TAM_COMENT+3];              /* Comentario 2 */
} TipoASP;
```

Figura 5.33: Estructura de datos utilizada para almacenar las primitivas de servicio de un archivo TTCN.

5.5.2.1.3.3 Esquema de conversión

La representación de la información de tipos y señales extraída del archivo TTCN, para poder ser usada en el sistema SDL, es la indicada en la Los tipos de datos pueden basarse en restricciones del conjunto posible de valores del tipo base (el suyo o, en el caso de tipos estructurados, el tipo base de cada campo). La herramienta procesa todas las posibles restricciones (Tabla 5.8), que incluyen la longitud máxima del valor, un rango de valores, etc. La forma de representar estas restricciones en SDL se muestra en la última columna de la tabla.

Tabla 5.7. Los tipos de datos simples se modelan con la estructura `syntype`; los tipos de datos estructurados, las PDUs y las ASPs se modelan como estructuras, mediante la construcción '`newtype <nombre_tipo> struct ...`'. Los tipos correspondientes a PDUs y ASPs se preceden del correspondiente prefijo '`PDU_`' o '`ASP_`'. En TTCN, los campos de las PDUs y las ASPs son todos opcionales; esta información no se puede incluir automáticamente porque depende de la especificación del protocolo. Además, un campo de una ASP puede declararse de tipo `PDU`, que denominaremos `meta-PDU` a partir de ahora para distinguirlo de las PDUs. Este tipo representa cualquier PDU válida. Se ha optado por construirlo, para las definiciones SDL, como una unión de todas las PDUs que se declaren.

Para los demás elementos (Parámetros de Pruebas, PCOs y TSOs) que constituyen la interfaz del Juego de Pruebas, se almacena una sinopsis de cada uno de ellos. Esta información puede utilizarse posteriormente en otras herramientas, como ocurre con los

SPICUM servicio de publicaciones

Parámetros de Pruebas, que se almacenan con el formato ya explicado en la sección 5.2.2.

Los tipos de datos pueden basarse en restricciones del conjunto posible de valores del tipo base (el suyo o, en el caso de tipos estructurados, el tipo base de cada campo). La herramienta procesa todas las posibles restricciones (Tabla 5.8), que incluyen la longitud máxima del valor, un rango de valores, etc. La forma de representar estas restricciones en SDL se muestra en la última columna de la tabla.

Tabla 5.7: Esquema de conversión a tipos SDL.

<table>
<tr><th colspan="2">Tipos de Datos Simples</th></tr>
<tr><th>Esquema</th><th>Ejemplo</th></tr>
<tr><td><pre>syntype <tipo> =
  <tipo_base>
    <Restricción>
endsyntype <tipo>;</pre></td><td><pre>syntype FU_STRING =
  Octet_String
    constants size(0:76)
endsyntype FU_STRING;</pre></td></tr>
<tr><th colspan="2">Tipos de Datos Estructurados, PDUs y ASPs</th></tr>
<tr><th>Esquema</th><th>Ejemplo</th></tr>
<tr><td><pre>newtype <tipo>
  struct
    <campo1> <tipo1>
    <campo2> <tipo2>
    ...
endnewtype <tipo>;</pre></td><td><pre>newtype PDU_PAGINGRESPONSE
  struct
    skipIndicator SkipIndicator;
    rRProtoDisc ProtocolDiscrim;
    msgType MsgType;
    spare4 B4;
    ciphKeySeqNum CiphKeySeqNum;
    mSClsmk2 MS_Clsmk2_lv;
    mobileId MS_Identity_lv;
endnewtype PDU_PAGINGRESPONSE;</pre></td></tr>
<tr><th colspan="2">Meta-PDU</th></tr>
<tr><th>Esquema</th><th>Ejemplo</th></tr>
<tr><td><pre>newtype PDU
  struct /*#UNION */
    <p1> <tipo_pdu1>
    <p2> <tipo_pdu2>
    ...
endnewtype PDU;</pre></td><td><pre>newtype PDU
  struct /*#UNION */
    pdu_auth_reject PDU_AUTH_REJECT;
    pdu_auth_reply PDU_AUTH_REPLY;
    pdu_auth_request PDU_AUTH_REQUEST;
    pdu_cc_alerting PDU_CC_ALERTING;
    pdu_cc_call_proc PDU_CC_CALL_PROC;
    pdu_cc_connect PDU_CC_CONNECT;
    pdu_cc_connect_ack PDU_CC_CONNECT_ACK;
endnewtype PDU;</pre></td></tr>
</table>

Tabla 5.8: Restricciones posibles en los tipos TTCN.

Restricción	Tipos posibles	Declaración en SDL
Lista de valores	Todos	`constants x, y, z;`
Rango de valores	`INTEGER`	`constants min:max`
Longitud	Cadenas	`constants size(longitud)`
Rango de longitudes	Cadenas	`constants size(min:max)`

SPICUM servicio de publicaciones

5.5.2.1.3.4 Uso de la herramienta

Invocando al generador *PRECCX* con los correspondientes archivos de reglas de producción, se obtiene el ejecutable de la herramienta *GenDef*. Este ejecutable puede ser invocado desde la línea de comandos según se indica en la Figura 5.34.

```
Uso:
    GenDef [-h] [-t] [-fnombre] < <fich.mp> [ >& <fich.err> ]
Opciones:
    -h: Muestra el mensaje de ayuda
    -t: Imprime los tipos encontrados y sus parámetros. Es sobre
        todo para depurar.
    -f: Lleva asociado el nombre, sin extensión, de los ficheros
        que se generarán. Por defecto usa 'salida' como 'nombre'.
    < <fich.mp>: Redirección para que la entrada estándar lea de
                 'fich.mp'.
    >& <fich.err>: Fichero opcional para la salida estándar.
```

Figura 5.34: Descripción del uso de la herramienta GenDef.

5.5.2.1.3.5 Salidas

La herramienta genera un fichero para cada uno de los grupos de elementos que constituyen la interfaz del Juego de Pruebas, con la extensión apropiada para cada caso (Tabla 5.9). Los comentarios incluidos en la declaración de cada elemento en el archivo TTCN también se incluyen en los ficheros de salida. La herramienta avisa de posibles inconsistencias en los módulos de entrada; por ejemplo, si algún tipo utilizado no está declarado.

Tabla 5.9: Ficheros generados por la herramienta GenDef.

Extensión	Tipo de Elementos	Contiene
`*.tad`	Tipos de datos simples y estructurados	Una sección para los tipos simples y otra para los estructurados.
`*.pdu`	PDUs	Un nuevo tipo por cada PDU.
`*.asp`	ASPs	Un nuevo tipo por cada ASP y el tipo PDU (correspondiente a la `meta-PDU` de TTCN)
`*.pco`	PCOs	Una línea para cada PCO con el nombre, el tipo y el papel a desempeñar.
`*.pic`	Parámetros de Pruebas	Una línea para cada Parámetro de Pruebas con información como el nombre del parámetro, el tipo de datos, etc.
`*.tso`	Operaciones	Para cada operación se incluye el nombre, el tipo del resultado a devolver y una descripción de qué debe hacer.

5.5.2.1.4 Comentarios finales

La herramienta *GenDef* ha sido desarrollada y utilizada para el desarrollo de Sistemas de Pruebas para DECT. La principal limitación que tiene es que sólo analiza los tipos de datos propios de TTCN. Como se puede ver de la Figura 5.27 a la Figura 5.29, hay

SPICUM servicio de publicaciones

secciones específicas para las declaraciones con ASN.1. Esta limitación no es tan grave, ya que, por ejemplo, la herramienta para desarrollo con TTCN de *Tau Suite* permite exportar a un fichero todas las declaraciones ASN.1 de un módulo; este fichero se puede incluir posteriormente en un sistema SDL, tal como indica [Z.105], que trata el uso combinado de SDL y ASN.1 ([FISC93], [HAUG93], [KARN93]). También hay que tener en cuenta que se tiende a declarar los tipos ASN.1 en un fichero aparte del Juego de Pruebas, lo que hace innecesario su extracción del módulo TTCN.

Los Juegos de Pruebas de DECT no hacen uso de las secciones ASN.1, por lo que la herramienta permite extraer completamente la interfaz. También se ha probado la herramienta con los Juegos de Prueba de Bluetooth, GPRS y UMTS y funciona correctamente, aunque con la limitación reseñada.

5.5.2.1.5 Mejoras

Sería fácil extender la funcionalidad de la herramienta para que considerara las declaraciones ASN.1. El problema que se plantea aquí es el de interpretar estas declaraciones, ya que se presentan en un único campo TTCN que debe ser recorrido para extraer la estructura de la declaración y pueden llegar a ser tan complejas como la mostrada en la Figura 5.35.

```
/* Tipo CphyTrchConfigReq */
SEQUENCE {
  activationTime SS_ActivationTime,
  ulconnectedTrCHList SEQUENCE (SIZE (0..maxTrCH)) OF
    SEQUENCE {
      trchid TransportChannelIdentity,
      ul_TransportChannelType SS_UL_TransportChannelType,
      transportChannelInfo CommonOrDedicatedTFS
    } OPTIONAL,
  ulTFCS TFCS OPTIONAL,
  dlconnectedTrCHList SEQUENCE (SIZE (0..maxTrCH)) OF
    SEQUENCE {
      trchid TransportChannelIdentity,
      dl_TransportChannelType SS_DL_TransportChannelType,
      transportChannelInfo CommonOrDedicatedTFS
    } OPTIONAL ,
  dlTFCS TFCS OPTIONAL
}
```

Figura 5.35: Tipo ASN.1 presente en un caso de prueba de UMTS.

En [VALL00] se ha diseñado un traductor de ASN.1, capaz de interpretar las declaraciones de tipos y valores y traducirlos al lenguaje SDL. Por tanto, para extender la funcionalidad de la herramienta *GenDef* se podría integrar el traductor en la herramienta o, de forma más sencilla, seguir el procedimiento aquí indicado:

1. Ampliar el analizador para que lea las secciones de declaraciones ASN.1 y genere uno o más archivos adicionales con las mismas.
2. Utilizar el traductor de ASN.1 a SDL para generar los tipos de datos correspondientes a las declaraciones anteriores.

Otra mejora sería introducir un analizador léxico externo como en [VALL00]. Esto evitaría tener que generar reglas gramaticales para cada *token* y cada literal, así como el incluir el *token* `espacio_en_blanco` en las reglas de la gramática.

SPICUM servicio de publicaciones

5.5.2.2 Generador de Codificador de la Interfaz

La salida generada por la herramienta *GenDef* es sólo uno de los dos pasos necesarios para lograr comunicar el Subsistema de Pruebas con el Subsistema Inferior. La información que circula entre ambos Subsistemas debe ser codificada como paso previo a su transmisión; en recepción, se debe proceder a su descodificación. La herramienta *GenCod* permite generar, de forma automática, el codificador que existe en el extremo del Subsistema Inferior.

Existen dos alternativas para la ubicación del codificador en el Subsistema Inferior. Son las siguientes:

a) Parte del sistema SDL: Las llamadas al codificador se realizan dentro del comportamiento del sistema SDL y están escritas en este lenguaje. Antes de transmitir una señal, se codifica la información que transmite; al recibir una señal, se invoca al descodificador.

b) En el Módulo Adaptador de los Protocolos: La codificación y descodificación se realizan, respectivamente, en las funciones que se encargan de la transmisión (`xOutEnv`) y de la recepción (`xInEnv`) de las señales que constituyen la comunicación externa del sistema SDL (ver sección 5.4.1.1).

La primera alternativa se ha evaluado en la codificación de las PDUs del Nivel de Red y de Enlace que son utilizadas en los Juegos de Pruebas de DECT (Apéndice I, Secciones I.3.1 e I.3.2). Sin embargo, la implementación generada se ha demostrado ineficiente, alcanzado una velocidad de ejecución muy baja, suficiente en el caso mencionado pero no para sistemas más exigentes.

Por tanto, se ha elegido la segunda opción como la más adecuada. El Módulo Adaptador de los Protocolos está implementado en C, lenguaje que permite una más cómoda manipulación de la información, tanto a la hora de codificarla como de descodificarla, que el lenguaje SDL.

El generador de codificadores se ha construido como una herramienta autónoma, ya que, como se verá, en algunas ocasiones es interesante el poder modificar la información extraída por *GenDef*. Las entradas al generador se indican en un fichero de texto donde se listan los archivos a emplear; estos archivos son los ficheros de tipos creados por *GenDef*. La información contenida en los archivos de tipos se almacena en memoria, en listas dinámicas, antes de generar el código correspondiente; la herramienta avisa de posibles inconsistencias en el conjunto de la información proporcionada.

La salida del generador son dos archivos, que contienen las correspondientes rutinas de codificación y descodificación:

- `Codec_SDL.h`: Declaración de funciones e identificadores de tipos.
- `Codec_SDL.c`: Implementación de las funciones.

La herramienta se ha construido adaptada a las características de los generadores de código del entorno de desarrollo *Tau Suite*. El generador de código C que posee este entorno para la notación TTCN permite el uso de dos posibles sintaxis de transferencia en la comunicación con el Subsistema Inferior: ASCII y BER/PER. El generador *GenCod* dispone de una versión para cada sintaxis de transferencia; tanto el proceso de generación como el formato de los archivos resultantes son diferentes según la sintaxis de transferencia empleada.

SPICUM servicio de publicaciones

Los siguientes apartados explican las características de las sintaxis de transferencia que admite *Tau Suite* y la implementación del generador *GenCod* en cada caso.

5.5.2.2.1 Sintaxis de transferencia

5.5.2.2.1.1 Sintaxis de transferencia ASCII

La sintaxis de transferencia ASCII que incluye *Tau Suite* está pensada sólo para el conjunto de tipos específicos de TTCN, por lo que no puede emplearse con las extensiones ASN.1. Esta sintaxis tiene la ventaja de ser legible en formato textual, pero el proceso es poco eficiente en velocidad, aparte de generar codificaciones poco compactas. Las reglas de esta sintaxis de transferencia se muestran en la Figura 5.36.

```
Message    ::= Value ';'
Value      ::= Sequence | SequenceOf | Base
SequenceOf ::=   TYPEID '{' ValueList '}'
               | TYPEID '[' ValueList ']'
               | '[' ValueList ']'
Sequence   ::=   TYPEID '{' ValueList '}'
               | TYPEID '[' ValueList ']'
               | '{' ValueList '}'
ValueList  ::= <empty> | Value { ,Value }
Base       ::= Boolean | Number | BString
               | Hstring | OString | CString
Boolean    ::= TRUE | FALSE
Number     ::= [- ][0-9][0-9]*
Bstring    ::= `'`[01]*`'`'B'
Hstring    ::= `'`[0-9A-F]*`'`'H'
Ostring    ::= `'`([0-9A-F][0-9A-F])*`'`'O'
Cstring    ::= TYPEID TextString
TYPEID     ::= [a-zA-Z][_0-9a-zA-Z]*
TextString ::= `"`[^"]*`"`
```

Figura 5.36: Sintaxis de transferencia ASCII[15].

```
MAC_DATA_REQ { 3, 1, pdu_information : { '1'B, '001'B, '00'B, '0'B, '1'B,
'000'B, '0'B, '000'B, '0'B, '001100'B, '0'B, '1'B, pdu_lce_page_response : {
'0'B, '111'B, '0000'B, , '01110001'B, 'BB039966F0'H, '06039966AA'H, ,  }, {
'11110000'B, '11110000'B, '11110000'B, , , ,  }, '3E4D'H } }
```

Figura 5.37: Ejemplo de codificación de un valor de tipo `MAC_DATA_REQ` según la sintaxis de transferencia ASCII.

La codificación del valor se realiza según las reglas establecidas para el tipo base, es decir, el tipo o construcción predefinidos del cual deriva el tipo del valor. Por ejemplo, un valor del tipo `LCE_HEADER` definido como

```
LCE_HEADER ::= BIT_3
BIT_3 ::= BITSTRING[3]
```

[15] Estas son las reglas de la sintaxis de transferencia que *Tau Suite* proporcionaba en su documentación hasta las últimas versiones; sin embargo, los valores de tipos `choice` se codificaban como `<nombre_del_campo_elegido> ':' <valor>`.

SPICUM servicio de publicaciones

se codificaría como un valor de tipo `BitString`. La Figura 5.37 muestra un ejemplo de codificación de un valor correspondiente a una primitiva `MAC_DATA_REQ`.

5.5.2.2.1.2 Sintaxis de transferencia BER/PER

El generador de código de TTCN construye funciones de codificación y descodificación basadas en las reglas BER [X.690] y PER [X.691], para el subconjunto de la notación ASN.1 utilizado en TTCN. La codificación PER soporta tres variantes: no alineada, alineada y sin relleno. Las dos primeras están orientadas a octetos, mientras que la tercera está orientada a bit. Para cada tipo ASN.1 se construye el siguiente par de funciones, siendo responsabilidad del programador la inicialización de los *buffers*:

```
UCFStatus Encode_<nombre_tipo> (tBuffer, GciValue*);
UCFStatus Decode_<nombre_tipo> (tBuffer, tLength*, GciValue**);
```

El uso de esta sintaxis de transferencia es especialmente interesante cuando la interfaz de los Juegos de Pruebas con el Subsistema Inferior está definida en ASN.1, como es el caso de UMTS. Un ejemplo de su uso de esta sintaxis de transferencia se muestra en la Figura 5.38, junto con su representación en la sintaxis de transferencia ASCII.

```
00 22 00 4A 02 DE 50 00 25 23 54 43 E0 0F F4 3F FF DD 21 0F 02 90 C0 AA 55 80
00 CC 5B 57 E2 BA 0F E8 3F A0 FE 80

PDU: { cellId: 0 , routingInfo: [rB_Identity: -1 ], tM_message:
[bCCH_BCH_Message: PDU: { message: { sfn_Prime: 0 , payload: [firstSegment: {
sib_Type: systemInformationBlockType5, seg_Count: 3 , sib_Data_fixed:
'00110101010001000011111000000000111111110100001111111
11111111101110100100001000011110000001010010000110000000101010100101011000000
00000000110011000101101101010111111000101011101000001111111010000011111110100
0001111111010'B } ]} } ]}
```

Figura 5.38: Ejemplo de codificación de un valor de tipo `RLC_TR_DATA_REQ` según la sintaxis de transferencia PER y su equivalencia en sintaxis de transferencia ASCII.

5.5.2.2.2 Implementación

5.5.2.2.2.1 Sintaxis de transferencia ASCII

La sintaxis de transferencia ASCII se ha utilizado en los Sistemas de Pruebsa de DECT y Bluetooth. Como entradas se utilizan los ficheros de tipos `*.tad`, `*.pdu` y `*.asp` (Figura 5.39) creados por *GenDef*. Al comienzo de la lista de archivos se puede indicar el nombre un archivo `*.xto` que contenga los nombres de aquellos tipos para los cuales no hay que generar las funciones de codificación.

```
DLC_FT
Tip_DLC_FTyPT_Gap_05s.tad
Tip_DLC_FTyPT_Gap_05s.pdu
Tip_DLC_FTyPT_Gap_05s.asp
```

Figura 5.39: Ejemplo de fichero de entrada para GenCod.

SPICUM servicio de publicaciones

Las tareas que realiza el generador una vez ha leído la información de los tipos de datos se dividen en tres grupos:

A) Generación de las cabeceras de los ficheros y declaración de las constantes y funciones.

B) Generación de las rutinas de codificación y descodificación de primer nivel. Estas rutinas actúan como multiplexores, determinando el tipo del objeto a procesar e invocando a la correspondiente rutina.

C) Generación de las rutinas de codificación y descodificación para cada tipo de datos (segundo nivel).

La Figura 5.40 muestra las tareas que se realizan en cada grupo y el orden temporal de las mismas. En el diagrama el término '*Copiar*' se refiere a que el generador se limita a escribir funciones o líneas de código que son siempre iguales; el término '*Generar*' identifica aquellas partes del código que son dependientes de la información presente en los ficheros de entrada, aunque la estructura de estas partes será igual para todo Sistema de Pruebas. A continuación se comentan algunas particularidades del código resultante.

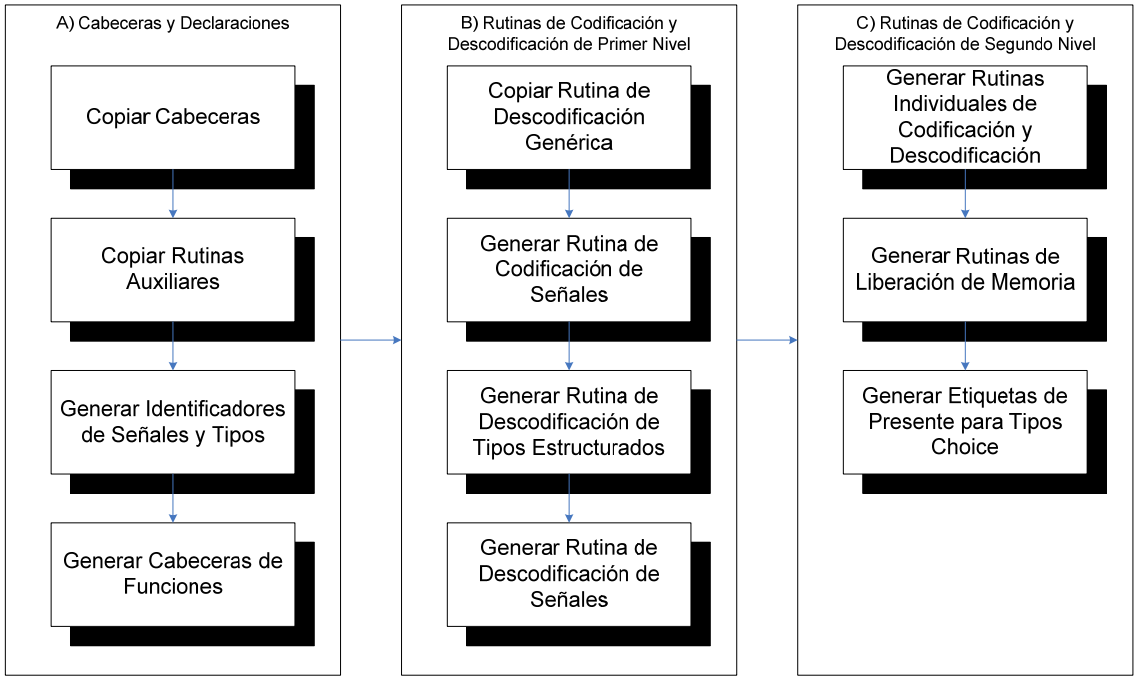


Figura 5.40: Esquema del flujo de tareas que realiza el generador GenCod para la sintaxis de transferencia ASCII.

Las rutinas auxiliares del primer grupo incluyen la construcción de la función `GeneralDecode__LookAhead`, que se utiliza para identificar el tipo base del siguiente elemento a descodificar en una cadena; esta función es siempre la misma. La rutina de descodificación, `GeneralDecode__Value`, aunque es genérica, se particulariza para poder tratar los tipos `CHOICE`.

La estructura de las funciones de codificación y descodificación de primer nivel correspondientes a las señales se muestra en la Figura 5.41. Estas dos funciones son las que se invocan, desde el Módulo Adaptador de los Protocolos, al recibir o enviar una señal. La rutina principal de codificación de señales, `Codif_Primitivas`, se limita a identificar qué señal es y a invocar la rutina correspondiente. La rutina principal de

SPICUM servicio de publicaciones

descodificación de señales, `Decod_Primitivas`, compara el siguiente elemento de la cadena a descodificar con los nombres de las señales; cuando encuentra uno que coincide, invoca a la función que descodifica la señal y, si no ha habido error, la introduce en el sistema SDL.

La rutina de primer nivel de descodificación de tipos estructurados, `Decod_Secuencia`, ya recibe como parámetro el tipo a descodificar y, al igual que en la codificación de señales, sólo hace falta invocar a la función apropiada.

```
int Codif_Primitivas(xSignalNode *S, char *OutStr)
{
  int error=FALSE;

  strcpy (OutStr,"");
  if (FALSE) { }
  else if (((*S)->NameNode)==<NOMBRE_SEÑAL_01>)
    error=Codif_ASP_<NOMBRE_SEÑAL_01> (
      &(((yPDP_<NOMBRE_SEÑAL_01>)(*S))->Param1 ), OutStr);
  else if (((*S)->NameNode)== <NOMBRE_SEÑAL_02>)
    error=Codif_ASP_<NOMBRE_SEÑAL_02> (
      &(((yPDP_<NOMBRE_SEÑAL_02>)(*S))->Param1 ), OutStr);
  ...
  else  {
    MENSAJE_ENTORNO("SEÑAL NO RECONOCIDA...");
    return (error=TRUE);
  }
  return (error);
}
```

```
int Decod_Primitivas(char * buf)
{
  xSignalNode SP;
  ...

  while (buf[pos++] != '{') {
    if (pos>=30) return (error=TRUE);
    if (!isspace(buf[pos-1])) tipo[ind++]= buf[pos-1];
  }
  tipo[ind]='\0';

  if (FALSE) { }
  else if (strncmp(tipo, "<NOMBRE_SEÑAL_01>", 16 ) ==0) {
    SP = xGetSignal(<NOMBRE_SEÑAL_01>, xNotDefPId, xEnv);
    error = Decod_ASP_<NOMBRE_SEÑAL_01> (buf,&pos,SP);
    if (error==FALSE)  {
      SDL_Output(SP xSigPrioPar(xDefaultSignalPrio), (xIdNode *)0);
      MENSAJE_ENTORNO("\nSeñal <NOMBRE_SEÑAL_01> enviada al SDL.");
    }
  }
  else if (strncmp(tipo, "<NOMBRE_SEÑAL_02>", 14 ) ==0) {
    SP = xGetSignal(<NOMBRE_SEÑAL_02>, xNotDefPId, xEnv);
    error = Decod_ASP_<NOMBRE_SEÑAL_02> (buf,&pos,SP);
    if (error==FALSE)  {
      SDL_Output(SP xSigPrioPar(xDefaultSignalPrio), (xIdNode *)0);
      MENSAJE_ENTORNO("\nSeñal <NOMBRE_SEÑAL_02> enviada al SDL.");
    }
  }
  ...
  return (error);
}
```

Figura 5.41: Estructura de las rutinas de primer nivel de codificación y descodificación de señales.

SPICUM servicio de publicaciones

El tercer grupo de tareas genera las rutinas necesarias para la codificación y la descodificación de cada tipo de datos. El esquema de las funciones generadas se muestra en la Figura 5.42.

```
int Codif_<NOMBRE_TIPO> (<NOMBRE_TIPO> *var_estruc, char *OutStr);
void *Decod_Struct_<NOMBRE_TIPO> (char *buf, int *pos, int *error);
void Libera_Mem_<NOMBRE_TIPO> (<NOMBRE_TIPO>* var_estruc);
```

Figura 5.42: Estructura de las cabeceras de las rutinas de codificación y descodificación de cada tipo de datos.

```
int Codif_<NOMBRE_META_PDU> (<NOMBRE_META_PDU> *var_pdu, char
                              *OutStr)
{
  int error= FALSE;

  switch(var_pdu->Present)  {
    case PRESENT_PDU_<NOMBRE_PDU_01>:
      error=Codif_PDU_<NOMBRE_PDU_01>
              (&(var_pdu->U.pdu_<nombre_pdu_01>),OutStr);
      break;
    ...
    default:
      MENSAJE_ENTORNO("\nERROR EN TIPO DE PDU");
      return (error=TRUE);
  }
  return (error);
}

       ____________________________________________________

void *Decod_<NOMBRE_META_PDU>(char *tipo, char *buf, int *pos, int
                              *error)
{
  void *var_ptr_aux;
  int tipo_pdu;
  PDU_NWK *var_pdu;

  var_pdu = (<NOMBRE_META_PDU> *) malloc(sizeof(<NOMBRE_META_PDU>));
  yDef_<NOMBRE_META_PDU> (var_pdu);
  tipo_pdu = parse(tipo);

  switch(tipo_pdu) {
    case T_PDU_<NOMBRE_PDU_01>:
      var_ptr_aux = Decod_Struct_PDU_<NOMBRE_PDU_01>
               (buf,pos,error);
      yAssF_PDU_<NOMBRE_PDU_01> ((var_pdu)->U.pdu_<nombre_pdu_01>,
               *(PDU_<NOMBRE_PDU_01> *) var_ptr_aux, XASS);
      Libera_Mem_PDU_<NOMBRE_PDU_01>
               ((PDU_<NOMBRE_PDU_01>*)var_ptr_aux);
      (var_pdu)->Present = PRESENT_PDU_<NOMBRE_PDU_01>;
      break;
    ...
    default:
      *error = TRUE;
      return (NULL);
  }
  return (void *) (var_pdu);
}
```

Figura 5.43: Estructura de las rutinas de codificación y descodificación de las `meta-PDU`.

SPICUM servicio de publicaciones

Para los tipos `CHOICE` hace falta añadir más información. Un tipo `CHOICE` almacena todas las alternativas posibles en una unión; para saber cuál de todas estas opciones se está usando, se incluye un campo `Present`. En el codificador hay que incluir los posibles valores que este campo puede tomar. Esta información se obtiene del fichero de cabeceras que genera la herramienta *Tau Suite* para los tipos usados en el sistema SDL. Un ejemplo de tipo `CHOICE` es el tipo `meta-PDU`, cuyas funciones de codificación y descodificación tienen una estructura similar a las de las señales (Figura 5.43).

Por último, comentar que, como se ha podido ver en las partes de código mostradas, todas las funciones del codificador devuelven un código de error si algo falla.

5.5.2.2.2.2 Sintaxis de transferencia PER

Esta versión del generador de codificador *GenCod* es una evolución del mismo para hacer uso de utilidades incorporadas al entorno de desarrollo con posterioridad a la primera versión. Inicialmente, en el Sistema de Pruebas para DECT se intentó definir la interfaz entre TTCN y SDL mediante ASN.1; sin embargo, esto no fue posible pues el uso de declaraciones de tipos en ASN.1 en SDL, en consonancia con la norma [Z.105], ocasionaba más inconvenientes de los que resolvía. Por dicho motivo se optó por utilizar la sintaxis de transferencia ASCII.

En versiones posteriores del entorno de desarrollo, se ha mejorado la integración entre SDL y ASN.1, y la herramienta ha incluido también un módulo de codificación capaz de utilizar las reglas BER y PER, esta última en sus diferentes versiones alineada (*aligned*) y no alineada (*unaligned*). Las rutinas de codificación y descodificación que hay que generar para una interfaz concreta se simplifican notablemente. La sintaxis de transferencia PER se ha usado en los Sistemas de Pruebas de UMTS, en concreto, en su variante no alineada.

La Figura 5.44 muestra los pasos en la ejecución del generador. Éste toma como entrada el módulo TTCN que describe el Juego de Pruebas. A partir de dicho módulo, se obtiene la lista de ASPs y los PCOs por los que circulan. Con la información obtenida, se escribe el fichero `Codec_SDL.h`, que contiene identificadores de los PCOs, el nombre en texto de todas las ASPs y las declaraciones de las funciones (Figura 5.45) de codificación (`EncodeASP`) y de descodificación (`DecodeASP`) invocadas por las funciones del Módulo Adaptador de los Protocolos `xOutEnv` y `xInEnv`, respectivamente.

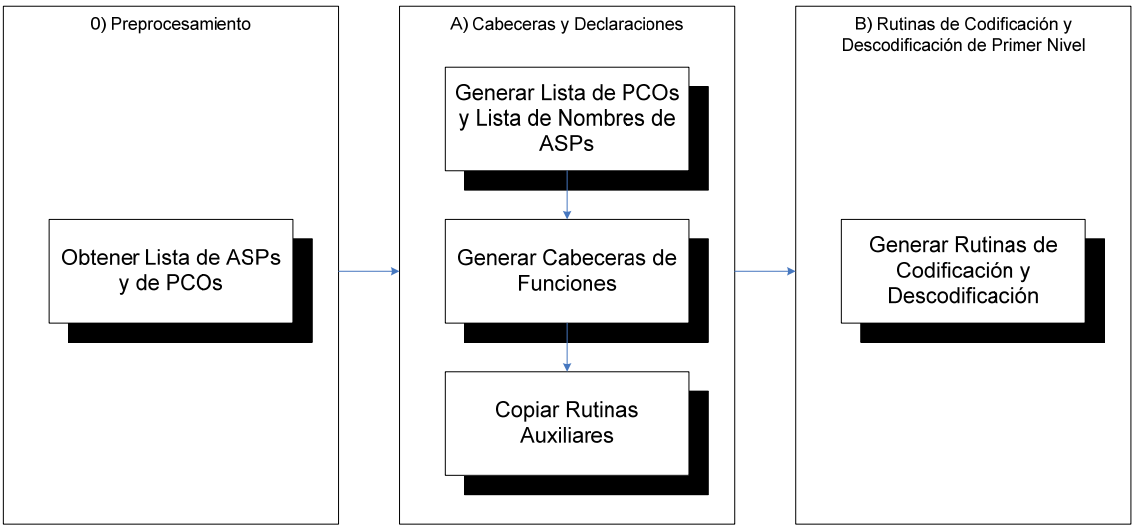


Figura 5.44: Esquema del flujo de tareas que realiza el generador GenCod para la sintaxis de transferencia PER.

SPICUM servicio de publicaciones

```
void DecodeASP(unsigned char *buffer, int longitud, tBuffer tBuf)
{
  xSignalNode signalIn;
  ...

  BufInitBuf(tBuf, bms_SmallBuffer);
  BufSetRule(tBuf, er_PER|er_Unaligned);
  BufInitWriteMode(tBuf);
  BufPutSeg(tBuf, buffer+1, longitud);
  BufCloseWriteMode(tBuf);

  switch (tipoASP) {
    case <NOMBRE_ASP_01>_ID:
      BufInitReadMode(tBuf);
      signalIn = xGetSignal(<NOMBRE_ASP_01>, xNotDefPId, xEnv);
      PER_DECODE(tBuf, (tASN1TypeInfo *)&yASN1_<nombre_asp_01>,
            (void *)&((yPDef_<NOMBRE_ASP_01> *)signalIn)->Param1);
      BufCloseReadMode(tBuf);
      viaList[0] = (xIdNode)xIN_CTRL;
      viaList[1] = (xIdNode)0;
      SDL_Output(signalIn xSigPrioPar(xDefaultSignalPrio), viaList);
      break;
    case <NOMBRE_ASP_02>:
    ...
  }
}
```

```
void EncodeASP(xSignalNode *signalOut, tBuffer tBuf,
               unsigned char *buffer, int *longitud)
{
  ...

  BufInitBuf(tBuf, bms_SmallBuffer);
  BufSetRule(tBuf, er_PER|er_Unaligned);

  if ((*signalOut)->NameNode == <NOMBRE_ASP_01>) {
    tipoASP = <NOMBRE_ASP_01>_ID;
    cellId = ((yPDef_<NOMBRE_ASP_01> *)(*signalOut))->Param1.cellId;
    BufInitWriteMode(tBuf);
    PER_ENCODE(tBuf, (tASN1TypeInfo *)&yASN1_<nombre_asp_01>,
          (void *)&((yPDef_<NOMBRE_ASP_01> *)(*signalOut))->Param1);
    BufCloseWriteMode(tBuf);
  }
  else if ( (*signalOut)->NameNode == <NOMBRE_ASP_02> )
  ...

  BufInitReadMode(tBuf);
  len = BufGetDataLen(tBuf);
  memcpy(buffer+4, BufGetSeg(tBuf, len), len);
  BufCloseReadMode(tBuf);

  ...

  buffer[3] = tipoASP;
}
```

Figura 5.45: Estructura de las rutinas de codificación y descodificación para la sintaxis de transferencia PER.

SPICUM servicio de publicaciones

5.5.3 Generador de Codificador/Descodificador para la Interfaz Aire (GenCodecAir)

Este apartado describe la herramienta *GenCodecAir* [PONC00], que genera automáticamente los procedimientos de codificación y descodificación de mensajes de los Niveles de Red y de Enlace. Estos procedimientos se han implementado en SDL, ya que han sido pensados para su integración en el modelo del Módulo de Protocolos. La herramienta está adaptada a la sintaxis de transferencia del Sistema DECT ([ETS 300 175-4], [ETS 300 175-5]). La necesidad de una herramienta de generación automática se puede deducir del hecho de que, por ejemplo, el Nivel de Red utiliza cerca de 20 PDUs, cada una de ellas con otros tantos campos que, a su vez, pueden estar modelados como tipos estructurados; en total, se requieren 109 rutinas codificadoras y otras tantas descodificadoras. El uso de esta herramienta permite modificar fácilmente este conjunto de rutinas si la definición o el número de los tipos varían.

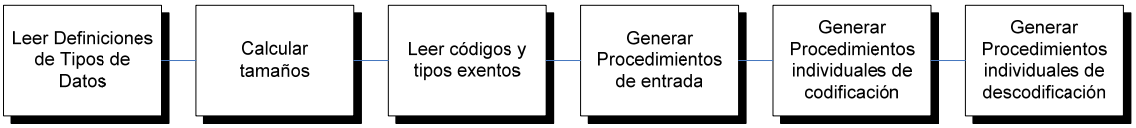


Figura 5.46: Etapas del generador GenCodecAir.

La Figura 5.46 muestra las etapas del generador. Como entradas se proporcionan los archivos que contienen las definiciones de los tipos de datos; estos ficheros habrán sido creados previamente por la herramienta *GenDef*: ficheros `*.asp`, `*.pdu` y `*.tad` (ver sección 0). Al almacenar dichas definiciones se calcula el tamaño que ocupan los valores de cada tipo una vez codificados. Primero se calcula el tamaño de los tipos base y, a partir de ellos, se va calculando el tamaño de los demás tipos en sucesivos recorridos de la lista; para este cálculo se tienen en cuenta las restricciones de rango admitidas en el lenguaje SDL (longitud, rango de valores y rango de longitudes). Si con la información disponible no se puede calcular el tamaño, la herramienta trata primero de hallar este tamaño en un fichero que haya proporcionado el usuario (extensión `*.usu`); si tampoco lo consigue, consulta al usuario. Como puede ser necesario utilizar esta herramienta en varias ocasiones sobre los mismos archivos, los tamaños introducidos por el usuario son almacenados para su consulta en futuras ejecuciones.

Después se leen los códigos numéricos de los Elementos de Información (IEI – *Information Element Identifier*) y de los mensajes del Nivel (PDUs). En ambos casos, si algún código no aparece en los correspondientes ficheros (`*.iei`, `*.tpu`), se genera automáticamente, con numeración secuencial, para futuras ejecuciones. Se puede indicar que un cierto tipo no sea procesado[16] incluyéndolo en el fichero de tipos exentos (fichero `*.xto`).

En el caso de los mensajes de Nivel de Red, la codificación se basa en el procedimiento `Conv_Octet_PDU_NWK`, que es el responsable de determinar el tipo de PDU a codificar e invocar al procedimiento correspondiente. Para cada PDU y cada tipo de datos se genera un procedimiento de codificación (ver Apéndice I, sección I.3.1). Un ejemplo se muestra en la Figura 5.47-a. Para simplificar el código[17] del codificador se emplean dos macros que permiten, respectivamente, comprobar si un campo está presente, invocando

[16] Por ejemplo, puede ocurrir que el tipo no se maneje o que se desee procesarlo manualmente.

[17] El formato textual, PR, del código SDL.

SPICUM servicio de publicaciones

en su caso a la correspondiente función de codificación, y comprobar el resultado de la codificación de un campo (Figura 5.48).

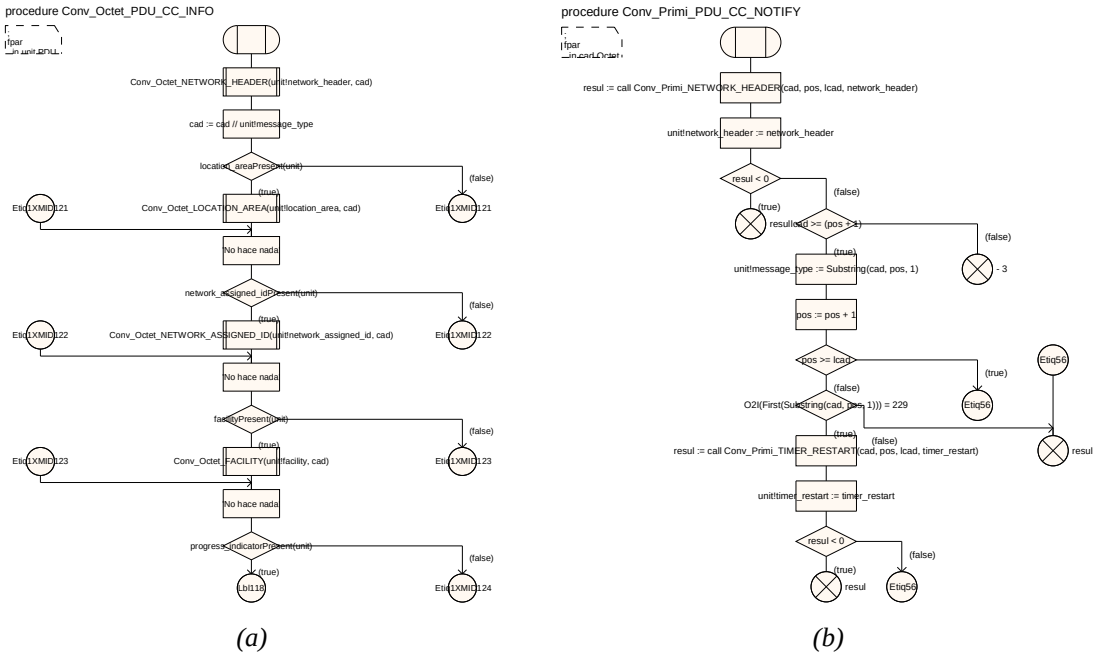


Figura 5.47: Ejemplos de los procedimientos (a) del codificador y (b) del descodificador.

En el descodificador, el procedimiento de entrada es `Conv_Primi_PDU_NWK`, que identifica el tipo de mensaje e invoca al procedimiento correspondiente; este procedimiento recorre cada campo de la PDU intentando descodificarlo. Al igual que en el codificador, cada PDU y cada tipo de datos dispone de un procedimiento de descodificación específico (ver Apéndice I, sección I.3.1). Estos procedimientos invocan a otros hasta que se alcanza un campo definido a partir de un tipo simple, que son los valores que se sabe descodificar. Un ejemplo de estos procedimientos se muestra en la Figura 5.47-b.

Como resultado de este procesamiento, se obtiene un fichero en formato textual (`PR`), que puede ser utilizado en el modelo SDL. Si la herramienta lo permite, se puede convertir este archivo a formato gráfico (`GR`) para un manejo más cómodo[18].

Al evaluar las prestaciones de los procedimientos generados mediante esta herramienta se ha observado que son considerablemente más lentos que si se hubieran realizado en el lenguaje C. El motivo es que el lenguaje SDL no dispone de mecanismos eficientes para realizar las tareas de codificación y descodificación, que requieren desplazamientos y concatenaciones a nivel de octeto y de bit. La única ventaja de este esquema es la posibilidad de observar gráficamente el comportamiento del codec, pero este argumento no es suficiente para recomendar el uso de la notación SDL en el codec.

[18] De ahí el aspecto en formato gráfico, GR, del código SDL del codificador, pues se genera tras expandir las macros.

SPICUM servicio de publicaciones

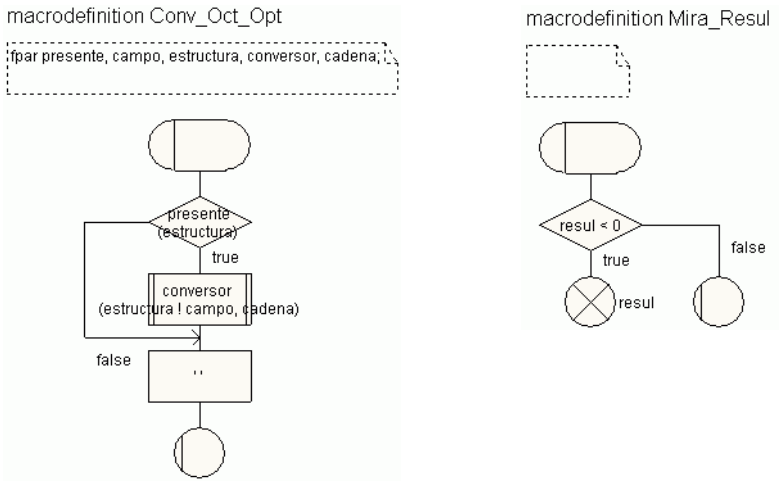


Figura 5.48: Macros generadas en el codificador.

5.6 Reglas de Nombrado

En el desarrollo de sistemas software el código resultante debe tener una apariencia lo más uniforme posible, para facilitar su legibilidad. Además, cuando se trata de un proyecto en que colaboran diferentes individuos surge también la necesidad de compartir módulos desarrollados por los demás. Esto conduce a fijar unas convenciones claras a la hora de nombrar los elementos que aparecen en el código, con objeto de evitar la duplicidad de nombres y que estos proporcionen información intuitiva sobre el elemento.

Esta necesidad, independiente del lenguaje de programación, ha llevado a definir una serie de reglas, que se han denominado Reglas de Nombrado, para cada elemento de un sistema SDL (señales, procesos, etc.). La Figura 5.49 muestra, de forma gráfica, ejemplos de uso de estas reglas de nombrado. Las convenciones que se han fijado se resumen en la Tabla 5.10[19].

Las reglas para cada uno de los elementos que pueden aparecer en un sistema SDL son las siguientes:

a) Estructura del Sistema:

- Sistemas y Tipos de Sistemas: Es la entidad principal y será el nombre del conjunto. Combinación de mayúsculas y minúsculas. Ej: `Nivel_DLC_PT`.
- Bloques, Procesos y Servicios: Los nombres de estas entidades deben ser significativos de su función, en mayúsculas y cortos, porque es posible que se referencien de modo habitual en forma de prefijos. Ej: `MAC_UTRAN`, `TM`, `PDCP`.
- Tipos de Bloques, Tipos de Procesos y Tipos de Servicios: Las reglas son las respectivas para los bloques, procesos y servicios.
- Procedimientos: Deben tener un nombre que represente su función. Estarán formados por palabras unidas o separadas por el carácter subrayado '_', teniendo la primera letra de cada palabra en mayúsculas. Se debe lograr ante todo que los procedimientos tengan nombres claros. Ej: `Get_AmHeader`, `CalculoPaging`, `Get_Timer_MRW`.

[19] Los ejemplos que aparecen en esta sección han sido tomados de los Sistemas de Pruebas de UMTS y DECT.

SPICUM servicio de publicaciones

Tabla 5.10: Ejemplo de nombres de los elementos de un modelo SDL.

Elemento	Ejemplo
Sistema	`Nivel_DLC_PT`
Bloques	`MAC_UTRAN, PDCP`
Procesos	`TM, UM_BMC`
Procedimientos	`Get_AmHeader, CalculoPaging`
Canales de Comunicación	`RLC_MAC, MAC_PHY`
Rutas de Comunicación	`MUXTX_AM, CTRL_TM`
Puntos de Acceso al Servicio	`AMSAP, CRLC, CTRL`
Señales	`CMAC_BMC_SCHEDULING_REQ, PHY_STATUS_IND`
Listas de Señales	`L..CRLCSAP..RLC..RRC, L..MUXTM..AM`
Tipos de Datos	`SystemInformationElem, Integer0_31`
Variables	`configReq, intModeCommand, operando`
Campos de una Estructura	`valorSFN, timer_Poll, numTF`
Constantes	`MAX_RB_NUMBER, MAX_SIB, RB_AM_15_RLC`
Parámetros	`bufferConfig, info`
Archivos	`Encode_TestData.spd, UMTS_RLC_Signals.sun`

b) Comunicación:

- Canales y Rutas de Comunicación: Son las vías a través de las cuales las entidades SDL se comunican; pueden ser unidireccionales o bidireccionales.

  i) Canales de Comunicación: Son los caminos por los que se comunican dos bloques o bien un bloque con el entorno del sistema. Los nombres de los canales se escriben con mayúsculas y siguen las siguientes reglas:

  (1) Entre un bloque y el entorno: el canal debe tener el mismo nombre que aparece en los Juegos de Pruebas; esta restricción facilita la comunicación entre los simuladores de SDL y de TTCN de la herramienta usada. Si no se hace así, se seguirá la regla para el caso de canal entre dos bloques. Ej: `CRLC`, `TM`.

  (2) Entre dos bloques:

  (a) Si el canal corresponde a la definición de un punto de acceso al servicio nombrado en la especificación, el canal recibirá el nombre de este punto de acceso, concatenado con el sufijo `SAP` (*Service Access Point*). Ej: `TMSAP`, `UMSAP`, `CMACSAP`.

  (b) En otro caso, el nombre del canal se formará de acuerdo a la expresión `<origen>_<destino>` siendo ambos el nombre de los puntos de acceso conectados de los bloques correspondientes; si es un canal bidireccional se toma como origen la entidad que se encuentre más arriba en la arquitectura del sistema. Ej: `RLC_MAC` (bidireccional), `MAC_PHY` y `PHY_MAC` (unidireccionales).

  ii) Rutas de comunicación: Son los caminos entre procesos y entre un proceso y el exterior del bloque en que se encuentra. El nombre, en mayúsculas, se formará de acuerdo a la expresión `<origen>_<destino>`. En el caso de una ruta bidireccional entre dos procesos, se tomará como origen el que se encuentre más arriba, de forma lógica, en la arquitectura del sistema; si no se puede determinar, se elegirá cualquiera de ellos. Ej: `MUXTX_AM`, `CTRL_TM`.

SPICUM servicio de publicaciones

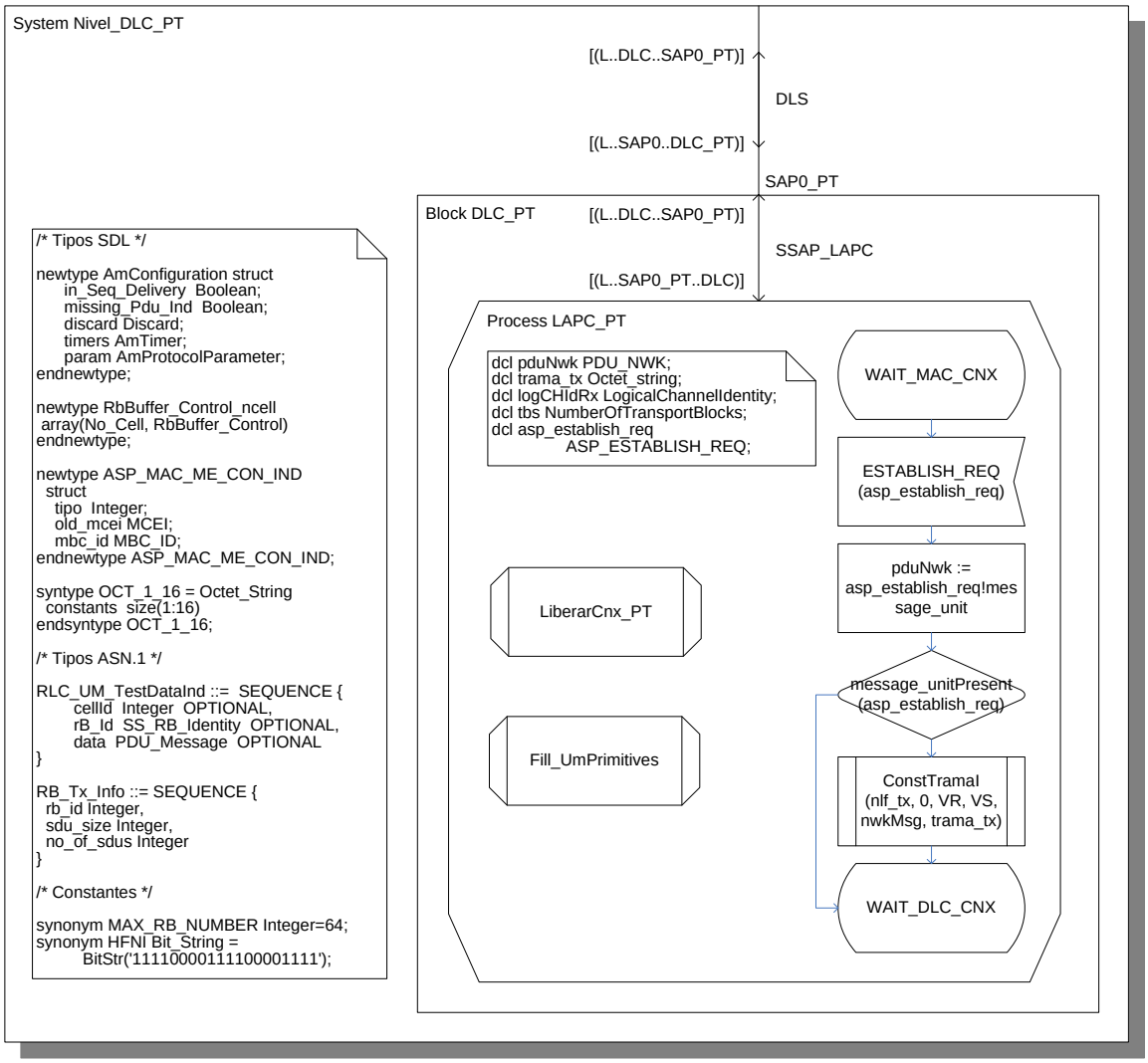


Figura 5.49: Ejemplos de nombres de elementos utilizados en los Sistemas de Pruebas.

- Puntos de Acceso al Servicio (SAP – *Service Access Point*): El nombre, en mayúsculas, será significativo del servicio lógico que se proporcione a través de ellos. Ej: `AMSAP`, `CRLC`, `CTRL`.
- Señales[20]: Las señales se escriben en mayúsculas, separando las palabras por caracteres de subrayado (‘_’). Para aquellas señales que vengan definidas en la norma, el nombre será el indicado en ésta[21]; éste es el caso de la mayoría de las señales que componen la interfaz del sistema con el entorno y las interfaces entre niveles. El resto de las señales, normalmente internas, debe tener un nombre significativo. Ej: `CMAC_BMC_SCHEDULING_REQ`, `CONFIG_REQ`, `RLC_AM_DATA_IND`, `PHY_STATUS_IND`.
- Listas de señales: Las señales se agrupan en listas para facilitar la lectura y manejo de un diagrama SDL. El nombre de estas listas se forma de acuerdo con la expresión `L..<sap>..<origen>..<destino>`. El nombre de las listas de

[20] Las primitivas de un protocolo se traducen en SDL a señales, y siguen el convenio aplicable a éstas.

[21] El carácter guión (‘-‘) que suele aparecer en el nombre de las señales debe sustituirse por el carácter subrayado (‘_’).

SPICUM servicio de publicaciones

señales se elige en el nivel de abstracción más alto en el que la lista se usa y se mantiene conforme se desciende en abstracción. El origen será el bloque o proceso donde se originen las señales de la lista; el destino será el bloque o proceso al que se encaminen. Si hay varias listas entre las mismas entidades, se puede utilizar el nombre de los puntos de acceso conectados para identificar cada una de las listas; la inclusión de este campo es opcional. Ej: `L..CRLCSAP..RLC..RRC`, `L..MUXTM..AM`, `L..PHY..MAC`.

c) Datos:

- Tipos de datos: Si los tipos de datos utilizados en las interfaces del sistema, o entre niveles, vienen ya definidos en algún documento, lo mejor es seguir el convenio utilizado. Habitualmente, el nombre de un tipo se forma concatenando una o más palabras, y poniendo en mayúsculas la primera letra de cada palabra; las siglas suelen escribirse todas en mayúsculas. En el caso de definir un rango de valores, por ejemplo enteros del 1 al 255, el nombre termina con este rango, anteponiendo a los límites inferior y superior un carácter subrayado ('_'); cuando el tipo defina no un rango sino un tamaño, se puede incluir este tamaño al final del nombre del tipo. Se ha probado también el uso de un prefijo, como puede ser una 'T', para distinguir los tipos de otros elementos; las ventajas no son tan relevantes, pues dificulta tanto la lectura como la escritura del código, obligando a pensar más en el convenio que fija la norma que en la semántica del tipo. Ej: `StringTrCh`, `SystemInformationElem`, `Integer_0_31`, `Octet_String_3`.
- Variables: Se escriben de forma similar a los tipos. El nombre puede estar formado por una o más palabras y empieza con minúsculas; la primera letra de las demás palabras se escribe en mayúsculas. Si una palabra es una sigla, puede aparecer en mayúsculas. Para facilitar la lectura puede utilizarse el carácter subrayado para separar alguna palabra. Ej: `configReq`, `intModeCommand`, `arrayContador`, `numRB`, `operando`.
- Campos de una Estructura: Siguen el convenio acordado para las variables, aunque no suelen estar formados por más de dos palabras. Si vienen definidos en las normas, se adoptará el convenio seguido en ellas. Ej: `valorSFN`, `timer_Poll`, `numTF`.
- Parámetros[22]: Siguen el convenio acordado para las variables, aunque lo normal es que estén formados por una sola palabra. Ej: `bufferConfig`, `info`.

Tabla 5.11: Extensiones asignadas para ficheros de un sistema SDL.

Archivo de Mensajes	`.pdu`	Archivo de Primitivas	`.asp`	Archivo de texto	`.txt`
Bloque	`.sbk`	Diagrama MSC	`.msc`	Macro	`.smc`
Operador	`.sop`	Paquete	`.sun`	Procedimiento	`.spd`
Proceso	`.spr`	Registro	`.log`	Servicio	`.ssv`
Sistema	`.ssy`	Subestructura	`.ssu`	Tipo de bloque	`.sbt`
Tipo de proceso	`.spt`	Tipo de servicio	`.svt`	Tipo de sistema	`.sst`
Tipos de datos	`.tad`				

[22] Los mensajes o unidades de datos se corresponden con parámetros de una primitiva, y siguen el convenio aplicable a estos.

SPICUM servicio de publicaciones

- Constantes: Para diferenciarlas de las variables, se aconseja escribir las constantes en mayúsculas, separando las palabras con caracteres de subrayado (‘_’). Ej: `MAX_RB_NUMBER`, `MAX_SIB`, `RB_AM_15_RLC`.

d) Organización del Modelo:

- Archivos: Los nombres deben ser significativos del contenido, mientras que la extensión (en minúsculas) del archivo expresará el tipo de contenido (Tabla 5.11). Los nombres de archivos no deben contener el carácter espacio (‘ ’), pues puede originar problemas en algunos sistemas de ficheros, así como otros caracteres no válidos. Ej: `Encode_TestData.spd`, `UMTS_RLC_Signals.sun`, `MUX_RX.ssv`, `AM.spr`.

5.7 Conclusiones

En este capítulo se han descrito una serie de herramientas que, junto con el entorno de desarrollo seleccionado, constituyen el conjunto de utilidades necesarias para poder llevar a la práctica los pasos descritos en la Metodología de Diseño descrita en el capítulo anterior. Las herramientas presentadas se clasifican en componentes genéricos de los Sistemas de Pruebas y generadores automáticos de interfaces

Parte de las herramientas desarrolladas son estáticas, como es el caso del Subsistema de Operación y Administración, mientras que otras deben ser particularizadas para cada Sistema de Pruebas específico. En este último grupo se ha intentado proporcionar la mayor automatización posible, aunque en ocasiones esto no es posible. Para el diseño de estas herramientas se ha puesto el máximo énfasis en la portabilidad a las plataformas Linux, Unix y Windows. La integración de estas herramientas en la Metodología quedará patente en capítulos posteriores, donde se describen implementaciones de Sistemas de Pruebas para distintas tecnologías.

SPICUM servicio de publicaciones

SPICUM
servicio de publicaciones

Capítulo

6

"'No quiero oír más', dijo Stilgar. Se giro y empezó a subir por entre las rocas en dirección al oasis al otro lado de la arena. Oyó a Leto seguirle. Poco después Leto lo rebasó y, girándose, dijo: '¿Has notado, Stil, qué hermosas están las chicas jóvenes este año?'"

Leto, Hijos de Dune

CAPÍTULO 6: ARQUITECTURA DE SISTEMAS DE PRUEBAS RADIO

Junto con las pruebas de conformidad de protocolos a que se someten los equipos de telecomunicación, existe una segunda vertiente que debe ser certificada antes de poder obtener el visto bueno por parte de las entidades reguladoras de cada país. Se trata de aquella parte del equipo que realiza la comunicación a nivel físico. En el caso de sistemas de comunicaciones móviles se utiliza un acceso radio, de ahí que se hable de pruebas de conformidad de radio.

Tradicionalmente, las pruebas de conformidad de radio y de protocolos se han considerado mundos distantes, ya que los ingenieros que trabajan en ellos no suelen tener formación en ambos campos. Mientras que las pruebas de protocolos están enfocadas hacia la comprobación de que las secuencias de mensajes intercambiadas entre dos entidades se realizan en el orden correcto y con la sintaxis adecuada, el objetivo de las pruebas radio es certificar el cumplimiento de aspectos como la compatibilidad radioeléctrica en transmisión y recepción.

Una diferencia esencial entre las pruebas de conformidad de radio y las pruebas de conformidad de protocolos es que las primeras requieren instrumentación específica capaz de llevar a cabo cierto conjunto de medidas radioeléctricas sobre la interfaz aire. Ejemplos de esta instrumentación son: generadores de señales, moduladores, osciloscopios, analizadores de espectros, etc. Esta instrumentación suele tener un coste bastante elevado, y, dado que habitualmente son necesarios varios equipos, ello conlleva que los Sistemas de Pruebas radio sean considerablemente más caros que los Sistemas de Pruebas de protocolos.

No obstante, la mayoría de los conceptos son compartidos entre ambos tipos de Pruebas. Al igual que en el caso de las pruebas de protocolo, las pruebas radio también son estandarizadas empleando una metodología similar, que incluye el mismo conjunto de documentos (Capítulo 2, Sección 2.2.5), con la única salvedad de que la metodología de pruebas sólo llega a escribir los escenarios y el comportamiento esperado del equipo en

SPICUM servicio de publicaciones

cada uno de ellos, pero esto tan sólo se hace al nivel del lenguaje natural. Falta, pues, la formalización del Juego de Pruebas Abstractas en la notación TTCN.

Dentro del marco de los organismos estandarizadores, durante los años noventa, se llevaron a cabo algunos intentos de formalización de las pruebas radio, pero se desecharon al no poder modelar formalmente todas las características que posee este tipo de pruebas. Sin embargo, un análisis de las especificaciones disponibles de pruebas radio induce a pensar que, si tal vez una formalización total no es posible, no cabe duda de que es posible modelar una gran parte de las interacciones y medidas que estas pruebas realizan.

En este capítulo se propone una metodología ([PONC07a], [PONC07b], [GOME01a]) para el diseño de Sistemas de Pruebas radio que acerque este campo al nivel alcanzado por las pruebas de protocolo[1]. La base se encuentra en la asimilación entre las pruebas de protocolo y las pruebas radio, viendo a estas últimas como una secuencia de mensajes intercambiados entre el Caso de Prueba y la Instrumentación. La formalización de las Pruebas radio incluye el modelado de estas interacciones mediante una interfaz de primitivas abstractas en TTCN, la cual permite la comunicación con el Subsistema Inferior. La información transportada por estas primitivas son mensajes abstractos, siendo el Subsistema Inferior el encargado de traducir estos mensajes a las secuencias de comandos propias de la instrumentación específica que incluya el Sistema de Pruebas.

En primer lugar, se explican las características de los Sistemas de Pruebas radio. Tras ello, se presentan los principios en los que se basa la metodología de diseño presentada en este capítulo. A continuación, se describe cómo adaptar los elementos de la arquitectura de los Sistemas de Pruebas de protocolos, y se expone cómo es posible modelar la Instrumentación de forma abstracta, detallando la interfaz entre la Instrumentación y el Caso de Prueba.

6.1 Visión General de los Sistemas de Pruebas Radio

Las pruebas radio son pruebas eléctricas de la interfaz aire, donde se determina si el EUT (*Equipment Under Test*)[2] emite y recibe dentro de los límites (frecuencia, potencia, banda, …) establecidos en las Especificaciones de Sistema. Para poder llevar a cabo el propósito de una prueba radio, el Sistema de Pruebas debe situar al EUT en un estado específico, en el cual se procede a realizar la medida pertinente. Las pruebas se suelen realizar en modo conducido[3], para que el EUT no se vea afectado por aspectos externos al escenario de la prueba. Entre las pruebas típicas se encuentran medidas de potencia, ancho de banda utilizado, desviación de la frecuencia portadora, calidad de la modulación, etc.

Un Sistema de Pruebas radio está formado por tres elementos (Figura 6.1). El primero de ellos es el Subsistema de Operación y Administración, que permite a un operador interactuar con el Sistema de Pruebas, visualizar resultados, etc. Cumple la misma

[1] Ideas similares, basadas en el trabajo aquí realizado, han sido también presentadas en [LOBA08].

[2] En las pruebas radio, la Implementación Bajo Prueba (IUT) se denomina Equipo Bajo Prueba (EUT).

[3] Las pruebas radio se pueden realizar de forma conducida o radiada. En el primer caso, las señales entre el Sistema de Pruebas y el Equipo Bajo Prueba se transmiten a través de un cable que conecta ambos sistemas; en el segundo caso, la comunicación se realiza a través del aire. Habitualmente se utiliza el primer método porque esto evita tanto las interferencias de sistemas ajenos a las pruebas como que la realización de las pruebas afecten a otros equipos.

SPICUM servicio de publicaciones

función que en el caso de Sistemas de Pruebas de protocolos. La interfaz con el operador de un Sistema de Pruebas Radio es equivalente a la ofrecida en los Sistemas de Pruebas de Protocolos: selección y ejecución de Casos de Prueba, edición de Parámetros de Pruebas, visualización de resultados, generación de Informes de Pruebas, etc. Adicionalmente, debe incluir un módulo de representación gráfica de las medidas realizadas, tanto en el dominio temporal como en el dominio de la frecuencia. La Figura 6.2 muestra un ejemplo de Interfaz de Operación empleado en un sistema comercial.

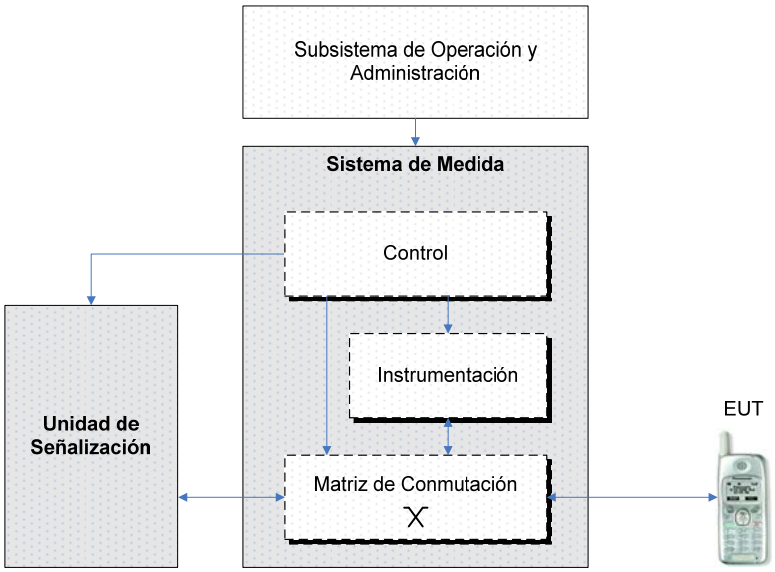


Figura 6.1: Elementos de un Sistema de Pruebas Radio.

El segundo elemento es el Sistema de Medida. Es el responsable global de la realización de la prueba y el encargado de realizar las medidas eléctricas. Desde un punto de vista conceptual una medida radioeléctrica (voltaje, corriente, frecuencia, …) en una Prueba radio es equivalente a la interpretación de la secuencia de bits que forma una trama en una Prueba de protocolos. El Sistema de Medida consta básicamente de instrumentación, controlada remotamente, capaz de realizar las medidas necesarias y un módulo de control, que coordina la realización de dichas medidas y la creación del escenario radioeléctrico. La instrumentación empleada incluye equipos como analizadores espectrales, osciloscopios, generadores de señal, etc. Las pruebas pueden necesitar también de circuitería adicional como filtros, amplificadores, etc. Las conexiones entre los distintos instrumentos dependen de la prueba que se ejecute. Tanto estas conexiones como las conexiones de la circuitería necesaria se realizan mediante una matriz de conmutación, que permite elegir el camino que debe recorrer la señal para cada prueba[4]. El control de la instrumentación puede hacerse de forma manual o a través de una interfaz de control remoto como, por ejemplo, RS232, VXI (*VME eXtensions for Instrumentation*), GPIB (*General Purpose Interface Bus*) o USB (*Universal Serial Bus*).

El tercer elemento se denomina Unidad de Señalización y es el encargado de llevar al EUT al estado requerido por cada Prueba; este elemento es dirigido por el módulo de

[4] La matriz de conmutación implementa caminos eléctricos que debe atravesar la señal para determinadas pruebas. Por ejemplo, caminos de atenuación o de reflexión.

SPICUM servicio de publicaciones

control del Sistema de Medida. En el caso de Sistemas de Pruebas de Protocolos, estos dos elementos se fusionan en uno solo, ya que el mismo elemento que realiza la señalización es el encargado de "medir" las PDUs recibidas, aunque estas medidas consistan en la recepción e interpretación de una trama.

Las Especificaciones de Prueba incluyen sólo el documento de la Estructura y Propósito de las Pruebas (TSS&TP). Es decir, no se genera el Juego de Pruebas en notación TTCN. Esto hace que cada suministrador desarrolle las pruebas radio con los lenguajes y notaciones que considera más adecuados. Alternativas habituales [BAÑO02] son tanto lenguajes de control de instrumentación, por ejemplo LabWindows/CVI (*C Virtual Instrument*) o HPVEE (*Hewlett-Packard Visual Engineering Environment*), como lenguajes de propósito general, por ejemplo C/C++.

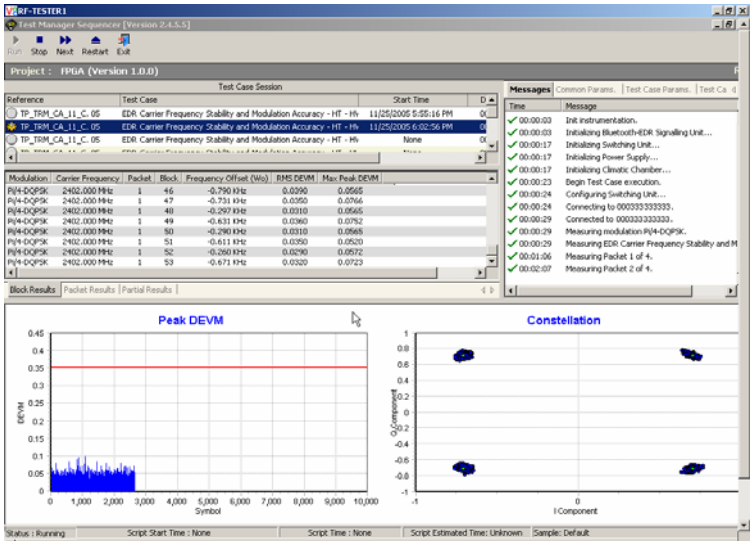


Figura 6.2: Interfaz de Operación del Sistema de Pruebas radio BITE [BITE].

6.2 Principios de Diseño

El enfoque de implementación de Sistemas de Pruebas Radio indicado en el apartado anterior de las pruebas radio tiene varias debilidades. Las más obvias son:

- Carencia de formalidad de las pruebas radio. Dado que cada suministrador implementa las pruebas a partir de una especificación en lenguaje natural, la validación de dichas implementaciones exige un mayor esfuerzo que en el caso de pruebas de protocolo, pues la posibilidad de un error es mayor al existir una mayor distancia entre la especificación en lenguaje natural y la implementación que entre ésta y los Juegos de Pruebas.
- Los entornos y lenguajes de diseño utilizados son distintos según si se implementan pruebas radio o pruebas de protocolos, con el consiguiente esfuerzo adicional de aprendizaje y de integración tanto con las demás herramientas de desarrollo como en el proceso productivo del suministrador.

Con objeto de superar estas limitaciones se propone utilizar los siguientes principios como parte integral del proceso de diseño de estos Sistemas de Pruebas:

SPICUM servicio de publicaciones

- Utilizar la misma arquitectura que en los Sistemas de Pruebas de protocolos. De esta forma es posible ofrecer al operador del Sistema de Pruebas una misma interfaz independiente del tipo de prueba (protocolo o radio) que esté ejecutando. Se reutilizan así los elementos que permiten la integración de los distintos Subsistemas. Esta arquitectura es sólo aplicable al Sistema de Medida.
- Formalizar las Pruebas Radio empleando la misma notación que para las pruebas de protocolos. TTCN es una notación diseñada ex profeso para el modelado de pruebas, por lo que ofrece construcciones de las que carecen otros lenguajes. Si esta notación se empleara para construir un ATS que se incluyera en las Especificaciones de Prueba estándares, se incrementaría la calidad de los Casos de Prueba. Por ejemplo, el modelado de estas pruebas sólo tendría que validarse una vez (el ATS), y no tantas como suministradores diferentes haya.
- Emplear los mismos entornos de desarrollo. Al utilizar la misma notación que para las pruebas de protocolo es posible reutilizar el mismo entorno, lo que hace que las interfaces que ofrece el código generado a partir de las pruebas sean las mismas. Esto posibilita el reutilizar no sólo otros componentes, como el Subsistema de Operación y Administración, sino también los generadores automáticos de interfaces utilizados en los Sistemas de Pruebas de protocolos. El diseñador no necesita aprender a trabajar en un nuevo entorno de desarrollo.

En conjunto, al poder reutilizar experiencia, herramientas y lenguajes, se consigue una reducción de costes y la mejora del proceso de diseño. En las siguientes secciones se describe cómo es posible aplicar estos principios.

6.3 Arquitectura de Sistemas de Pruebas Radio

La Arquitectura de los Sistemas de Pruebas de protocolos se puede emplear para el Sistema de Medida de los Sistemas de Pruebas radio de la forma en que se muestra en la Figura 6.3. Los distintos componentes se explican a continuación.

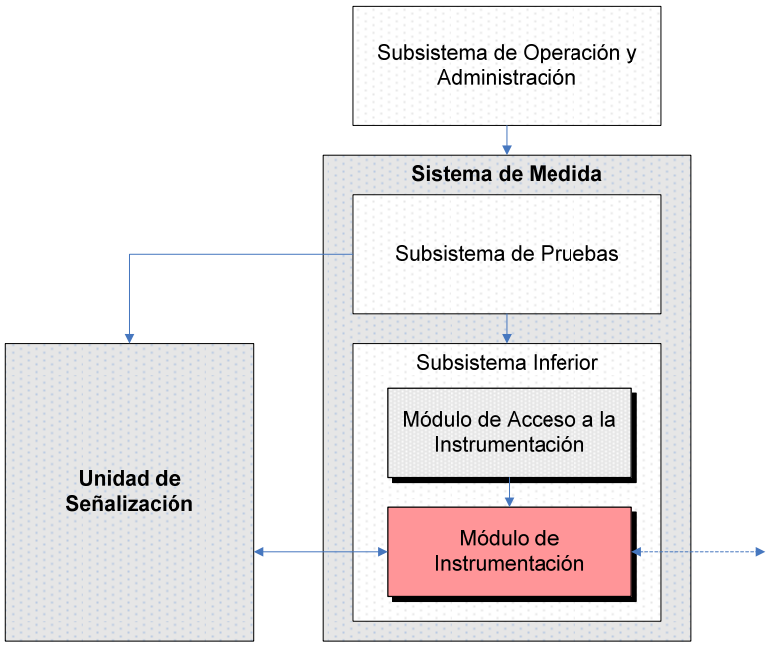


Figura 6.3: Arquitectura de un Sistema de Pruebas Radio.

SPICUM servicio de publicaciones

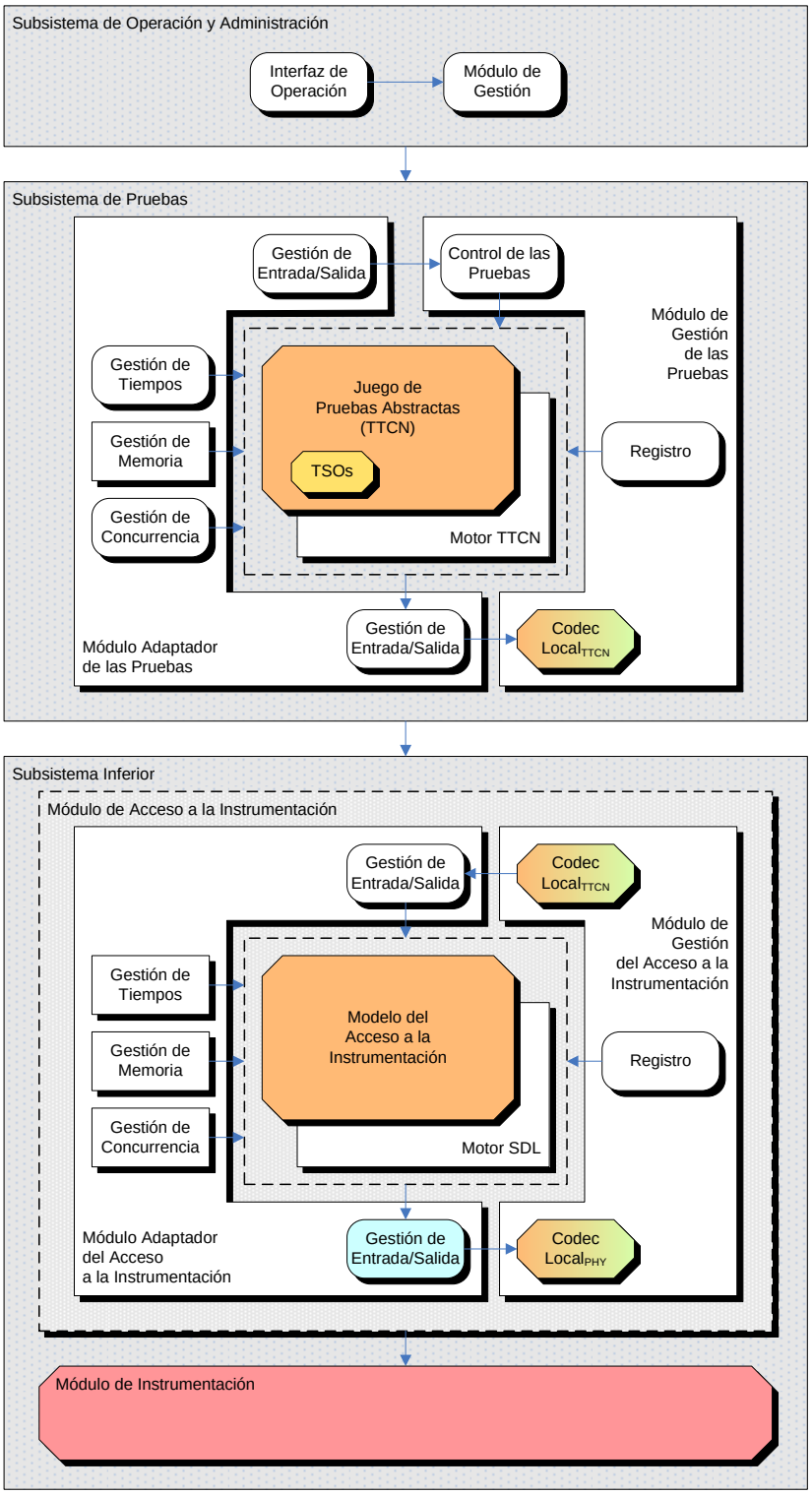


Figura 6.4: Componentes específicos de un Sistema de Pruebas Radio.

Como se ha mencionado previamente, el Subsistema de Operación y Administración ofrece una funcionalidad equivalente a la incluida en los Sistemas de Pruebas de protocolos. La diferencia es que la visualización de resultados se realiza mediante la representación gráfica de las medidas, en lugar de mediante secuencias de mensajes. El Subsistema de Operación y Administración se ha separado del Sistema de Medida para

SPICUM servicio de publicaciones

recalcar el hecho de que son el Subsistema de Pruebas y el Subsistema Inferior quienes realizan las Pruebas, de forma equivalente al caso de Sistemas de Pruebas de protocolo.

El Sistema de Medida está formado por el Subsistema de Pruebas y el Subsistema Inferior. El propósito de ambos Subsistemas es equivalente al caso de los Sistemas de Pruebas de protocolos:

- El Subsistema de Pruebas controla la ejecución de las Pruebas y realiza la secuencia de interacciones (estímulos y eventos) necesaria para ello; controla tanto la Unidad de Señalización como la Instrumentación. Al utilizar la notación TTCN para modelar los Casos de Prueba, la estructura de este Subsistema no varía con respecto a los Sistemas de Pruebas de protocolos.
- El Subsistema Inferior proporciona acceso a la interfaz aire del EUT. El Subsistema Inferior agrupa dos módulos
    - Módulo de Instrumentación: incluye todos los instrumentos necesarios para las Pruebas más la matriz de conmutación.
    - Módulo de Acceso a la Instrumentación: es responsable de adaptar la comunicación entre la Instrumentación y el Subsistema de Pruebas. Realiza el papel del Módulo de Protocolos en los Sistemas de Pruebas de protocolos.

Al modelar las Pruebas en TTCN y emplear la misma arquitectura, es posible reutilizar la mayoría de los componentes de la Arquitectura de Sistemas de Pruebas de protocolos. La Figura 6.4 muestra los elementos que componen un Sistema de Pruebas radio con el mismo nivel de detalle utilizado en el Capítulo 3, donde se describió la arquitectura de los Sistemas de Pruebas de protocolos, y resalta los componentes que son específicos de dichos Sistemas de Pruebas radio. Se pueden observar algunas diferencias entre ambas arquitecturas.

No se incluyen los codificadores extremo-extremo que aparecían en el Juego de Pruebas Abstractas y el Modelo Abstracto de Protocolos. Estos codificadores no son necesarios ya que el Sistema de Medida no se comunica con el EUT; la comunicación se realiza entre el Subsistema de Pruebas y la Instrumentación.

El Codec $Local_{TTCN}$ no hay que implementarlo, pues el mismo que el empleado en las pruebas de protocolos. Requiere, eso sí, la adaptación al conjunto de primitivas y mensajes empleados en la interfaz entre el Subsistema de Pruebas y el Subsistema Inferior.

Sin embargo, hay que construir el Codec $Local_{PHY}$, pues debe adaptarse a la interfaz lógica ofrecida por los instrumentos; como se verá más tarde, este codificador es reutilizable entre diferentes Sistemas de Pruebas radio. Igual ocurre con el elemento de Gestión de Entrada/Salida en la frontera inferior del Módulo de Acceso a la Instrumentación. Este elemento oculta la interfaz física ofrecida por los instrumentos; por ello, también es reutilizable y sólo debe ser construido una vez.

El Modelo del Acceso a la Instrumentación sustituye al Modelo Abstracto de Protocolos, ya que ahora sólo se utiliza como intermediario entre el Subsistema de Pruebas y la Instrumentación. No se ha indicado un lenguaje específico para modelar su comportamiento; este punto se trata en la Sección 6.5.1, una vez descrito cómo se modelan los Juegos de Pruebas. Del mismo modo, el Motor denota el conjunto de librerías que implementan la semántica del lenguaje que se utilice. Los Módulos Adaptador y de Gestión del Acceso a la Instrumentación son los mismos que los

SPICUM servicio de publicaciones

utilizados en los Sistemas de Pruebas de protocolos; se les ha cambiado el nombre para reflejar el uso de la Instrumentación.

La Unidad de Señalización es un elemento que puede considerarse una entidad independiente en sí misma. Con objeto de poder automatizar la ejecución de las Pruebas, debe ofrecer una interfaz que permita su control desde el Subsistema de Pruebas. Realiza todos los niveles de protocolos necesarios para la señalización entre el Sistema de Pruebas y el EUT, con objeto de situar a éste en el estado adecuado para cada Prueba.

La Unidad de Señalización se puede construir haciendo uso de los Juegos de Pruebas de protocolos y su correspondiente Subsistema Inferior. Los Casos de Prueba ya incorporan las secuencias de pasos necesarias para situar al EUT en la mayoría, si no todos, de los estados iniciales de las pruebas radio. Se puede eliminar el código superfluo y extraer las secuencias de inicialización y cierre necesarias, generando un nuevo Subsistema de Pruebas de protocolos. La selección y ejecución de las secuencias adecuadas se controlaría desde el Subsistema de Pruebas radio a través de una interfaz específica. A lo largo de este capítulo no se volverá a tratar este punto, ya que su implementación es relativamente simple.

6.4 Juegos de Pruebas Abstractas Radio

Un Caso de Prueba radio se divide en las mismas etapas que los Casos de Prueba de protocolos (prólogo, cuerpo y epílogo), teniendo cada una de ellas el mismo significado. Los Casos de Prueba radio tienen una apariencia más lineal que los Casos de Prueba de protocolos, ya que una vez situado el EUT en el estado deseado se realiza la medida y termina la Prueba; las posibles ramas alternativas de comunicación con el EUT que se dan en las Pruebas de protocolo quedan ocultas en la Unidad de Señalización.

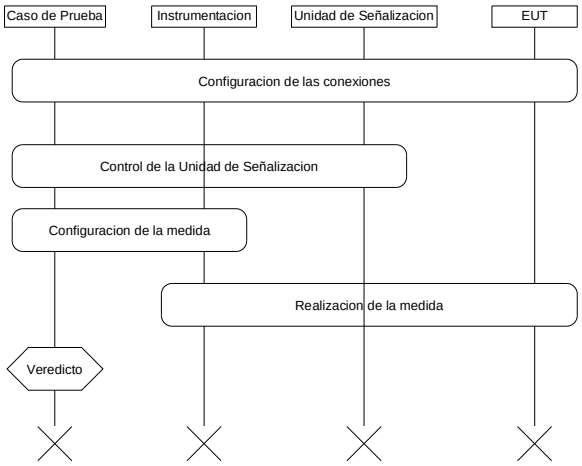


Figura 6.5: Secuencia de acciones típica en un Caso de Prueba radio[5].

En un Caso de Prueba radio, visto a alto nivel, se realizan las siguientes acciones (Figura 6.5):

[5] Cuando una línea cruza por delante de una de las cajas, indica que la entidad no participa en esta actividad.

SPICUM servicio de publicaciones

1. Creación del escenario de la Prueba. Engloba los siguientes aspectos:
   a. Configuración de las conexiones entre la instrumentación, la Unidad de Señalización y el EUT.
   b. Control de la Unidad de Señalización.
2. Configuración de la medida a realizar.
3. Realización de la medida.
4. Emisión del veredicto.

Una medida está caracterizada por:

- Tipo de medida,
- Parámetros asociados al tipo de medida y
- Conexionado (configuración eléctrica) de los elementos del Sistema de Pruebas.

Dado que los Casos de Prueba son abstractos, para que reflejen fielmente el Propósito de la Prueba es necesario modelar de forma abstracta el concepto de medida. Esta abstracción implica dos aspectos: la definición de forma abstracta de la instrumentación necesaria para dicha medida, con objeto de especificar tanto la semántica de la medida como su configuración eléctrica y la comunicación con los demás elementos del Sistema de Pruebas, con objeto de fijar los parámetros de la medida a realizar. Ambos aspectos están relacionados, fundamentalmente, con la Instrumentación empleada.

El procesado de señal que puede requerir una Prueba radioeléctrica fuera de la configuración y realización de la medida, como por ejemplo el cálculo de la desviación de la frecuencia de portadora, no se modela en notación TTCN, ya que esta notación no es adecuada para desarrollar algoritmos matemáticos. Estos algoritmos se modelan en funciones externas, que son invocadas desde los Casos de Prueba.

Los apartados 6.4.1 y 6.4.2 describen cómo es posible incorporar y emplear un modelo abstracto de la Instrumentación dentro de los Casos de Prueba.

6.4.1 Modelado de la Instrumentación

Para modelar de forma abstracta la Instrumentación se va a definir el concepto de Instrumento Virtual (IV). Un Instrumento Virtual es un elemento capaz de realizar una o varias medidas, cada una de las cuales puede ser configurada mediante cero o más parámetros y que dispone de uno o más conectores de entrada y/o salida.

La formalización de la Instrumentación comienza definiendo un conjunto de Instrumentos Virtuales que sea comúnmente aceptado en pruebas radioeléctricas. Los Juegos de Pruebas harán uso de los Instrumentos Virtuales de esta lista. La matriz de conmutación se puede ver como un conjunto de caminos eléctricos, cada uno de los cuales se puede modelar como un Instrumento Virtual[6]. Para modelar un Instrumento Virtual hay que especificar las siguientes características del mismo:

1. Puertos eléctricos (conexiones) y sentido (entrada o salida).

[6] Otra opción es modelar la matriz de conmutación como un Instrumento Virtual especial. Esta solución es menos general, ya que la funcionalidad de este Instrumento Virtual dependería del Juego de Pruebas, pues la matriz es diferente para cada Sistema de Pruebas, y no podría modelarse como un Instrumento Virtual genérico. A la hora de construir el Sistema de Pruebas la implementación de esta matriz de conmutación en instrumentación física no podría hacerse de forma automatizada.

SPICUM servicio de publicaciones

2. Tipos de medidas que puede realizar (semántica).
3. Parámetros configurables para cada tipo de medida.

Al igual que con los Instrumentos Virtuales, es necesario enumerar tanto los posibles tipos de medidas como el conjunto de parámetros configurables de cada una de ellas, y que estos listados sean aceptados tanto por los especificadores de pruebas como por los suministradores de los Sistemas de Pruebas. Una posible modelización de la Instrumentación es la definida por SCPI (*Standard Commands for Programmable Instrumentation*) [SCPI99], que especifica un conjunto de comandos para el control de instrumentación.

La siguiente cuestión es dónde definir los Instrumentos Virtuales. Esta definición puede realizarse en el mismo documento donde se especifica el Método de Pruebas, bien mediante una especificación completa de cada Instrumento Virtual o referenciando a algún otro estándar donde dichos Instrumentos Virtuales hayan sido especificados. Esta definición se realizaría en lenguaje natural. Un Instrumento Virtual puede ser, por ejemplo, un medidor de potencia, y sus parámetros configurables podrían ser la banda de frecuencias a medir, el tipo de medida (potencia media, potencia de pico, …), la duración de la medida, etc. La opción más satisfactoria sería poder modelar estos Instrumentos dentro del propio módulo TTCN, pero la semántica actual de TTCN no permite esta vía ya que no se pueden declarar elementos externos al Juego de Pruebas; en el apartado 6.4.1.1 se comenta cómo se podría aplicar esta alternativa.

Antes de realizar una prueba hay que fijar la configuración de la medida, es decir, conectar la Instrumentación, la Unidad de Señalización y el EUT entre sí de forma adecuada. Este conexionado se realiza de forma directa o, si se necesita circuitería adicional, a través de la matriz de conmutación. Las conexiones a realizar se pueden especificar dentro del código TTCN de cada Caso de Prueba. Dado que requiere comunicarse con el Subsistema Inferior, la indicación de qué conexiones son necesarias se modelará como una primitiva que indique los puertos a conectar. Esta primitiva se describe en la sección 6.4.2.

A la hora de implementar las Pruebas, es necesario realizar los Instrumentos Virtuales mediante instrumentación física. ¿Qué ventaja proporciona, por tanto, esta abstracción frente al método actual donde las Especificaciones de Prueba definen las medidas a realizar pero no se modela la instrumentación? Entendemos que la ventaja que se obtiene es una mayor formalización de la instrumentación requerida por las Pruebas, lo cual permite automatizar en mayor medida la construcción del Sistema de Pruebas. En concreto, al definir los Instrumentos Virtuales los Juegos de Pruebas harán uso de ellos, referenciándolos por nombres identificativos de los mismos. Al generar el Subsistema de Pruebas, es posible interactuar con la Instrumentación a través de un módulo que sea el mismo para todos los Juegos de Pruebas, lo que hemos denominado el Módulo de Acceso a la Instrumentación.

6.4.1.1 Modelado mediante Componentes Paralelos de Prueba

Una alternativa interesante para poder modelar los Instrumentos Virtuales formalmente dentro del propio Juego de Pruebas es hacer uso de los Componentes Paralelos de Prueba (PTC). Cada Instrumento Virtual se modelaría como un PTC donde se definirían los puertos del Instrumento Virtual como PCOs del Componente de Prueba. Esto permitiría crear Instrumentos Virtuales, conectarlos y destruirlos mediante las propias construcciones de TTCN (`create`, `start`, `stop`, `kill` y `connect`). Al recibir un

SPICUM servicio de publicaciones

mensaje desde el Caso de Prueba, el PTC reenviaría dicho mensaje a la Instrumentación Virtual. Esto requiere definir un nuevo PCO en el PTC para la comunicación con la Instrumentación aparte de los puertos propios del mismo[7].

Esta alternativa requiere modificar la semántica de TTCN, pues no está contemplado que un Componente de Prueba se realice por una entidad ajena al Juego de Pruebas. Además, la conexión entre PCOs de distintos Componentes de Prueba debe realizarse externamente, lo que obligaría a tener esto en cuenta en el código generado, invocando a mecanismos externos en estos casos. Una opción sería ampliar la semántica de los Componentes Paralelos de Prueba con una palabra reservada, por ejemplo `extern`, que denotara que este Componente se realiza fuera del Juego de Pruebas. Los cambios necesarios en los generadores de código serían mínimos. Simplemente tendrían que identificar si el PTC es externo o no, y, en el primer caso, remitir las operaciones sobre el PTC y toda la comunicación a un módulo externo que podría ser, por ejemplo, el propio SUT. Es, como se observa, un cambio pequeño pero que permitiría emplear toda la potencia de modelado de TTCN para formalizar las pruebas radio, ya que evitaría tener que definir los Instrumentos Virtuales de forma externa al Juego de Pruebas.

6.4.2 Interfaz con la Instrumentación

En este apartado se describe la interfaz ([PONC07a]) que permite controlar la Instrumentación desde el Juego de Pruebas. Esta interfaz está compuesta por cinco primitivas. Todas ellas son confirmadas, lo que permite detectar cualquier error en la Instrumentación de forma inmediata. La comunicación con el Subsistema Inferior se puede realizar a través de un único PCO o de varios.

Tabla 6.1: Primitivas de comunicación con el Módulo de Acceso a la Instrumentación.

Primitiva	Parámetros
`INIT_INSTRUMENT_REQ`	`(INSTRUMENT id, ListPORTS lp)`
`INIT_INSTRUMENT_RSP`	`(INTEGER cod_error, IA5String cad_error)`
`CONNECT_REQ`	`(INSTRUMENT id1, PORT port1, INSTRUMENT id2, PORT port2)`
`CONNECT_RSP`	`(INTEGER cod_error, IA5String cad_error)`
`SET_PARAMETER_REQ`	`(INSTRUMENT id, INSTPAR par, PARVAL val`
`SET_PARAMETER_RSP`	`(INTEGER cod_error, IA5String cad_error)`
`GET_PARAMETER_REQ`	`(INSTRUMENT id, INSTCOM com, COMVAL val)`
`GET_PARAMETER_RSP`	`(IA5String measure, INTEGER cod_error, IA5String cad_error)`
`STOP_INSTRUMENT_REQ`	`(INSTRUMENT id)`
`STOP_INSTRUMENT_RSP`	`(INTEGER cod_error, IA5String cad_error)`

La Tabla 6.1 resume las primitivas definidas. La Figura 6.6 muestra el uso de estas primitivas en el flujo lógico del Caso de Prueba. La semántica de estas primitivas es la siguiente:

a) `INIT_INSTRUMENT_REQ (INSTRUMENT id, ListPORTS lp)`

   Equivale a una creación del Instrumento Virtual. Hace que se active la instrumentación específica que lo realiza. El primer parámetro, `id`, determina el

[7] Todos los PCOs de este tipo usados por todos los PTCs que modelaran Instrumentos Virtuales se pueden asociar al mismo PCO del Subsistema de Pruebas.

SPICUM servicio de publicaciones

Instrumento Virtual. El segundo parámetro, `lp`, contiene el nombre de los puertos de este Instrumento Virtual; se utiliza para asociar el nombre que utiliza el ATS a cada puerto. Estos nombres aparecen en el mismo orden que en la especificación del Instrumento Virtual, de forma que la pareja nombre-puerto pueda ser asociada automáticamente.

b) `CONNECT_REQ (INSTRUMENT id1, PORT port1, INSTRUMENT id2, PORT port2)`

Es la primitiva empleada para establecer las conexiones entre los distintos Instrumentos Virtuales. Los dos primeros parámetros (`id1`, `port1`) determinan el puerto del primer Instrumento Virtual; los dos siguientes parámetros (`id2`, `port2`) se refieren al segundo Instrumento Virtual.

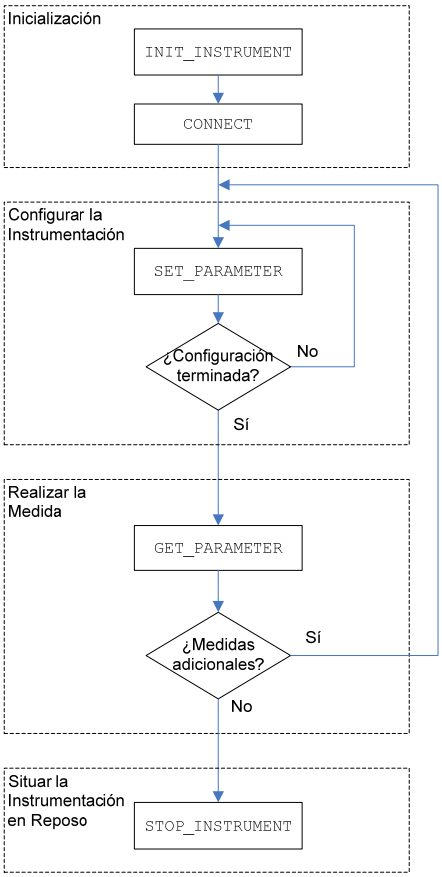


Figura 6.6: Flujo lógico del uso de las primitivas de la interfaz.

c) `SET_PARAMETER_REQ (INSTRUMENT id, INSTPAR par, PARVAL val)`

Configura el Instrumento Virtual `id` fijando el valor `val` en el parámetro `par`.

d) `GET_PARAMETER_REQ (INSTRUMENT id, INSTCOM com, COMVAL val)`
`GET_PARAMETER_RSP (MEASUREMENT measure, INTEGER cod_error, IA5String cad_error)`

Solicita que el Instrumento Virtual `id` ejecute el comando `com` con el valor `val`. Este comando será una medida a realizar, por lo que la respuesta debe incluir el valor de la medida realizada.

e) `STOP_INSTRUMENT_REQ (INSTRUMENT id)`

SPICUM servicio de publicaciones

Devuelve al Instrumento Virtual al estado de reposo en el que se debe encontrar al inicio de cada Caso de Prueba. Esto incluye la desconexión de los puertos del Instrumento Virtual.

La Tabla 6.1 también incluye las primitivas de respuesta. Como se observa, todas ellas incluyen dos parámetros comunes: un código (`cod_error`) que indica el tipo de error que se ha producido, si ha ocurrido alguno, y una descripción textual (`cad_error`) del mismo. Si se deseara poder enviar varias primitivas sin esperar confirmación, tan sólo sería necesario incluir un parámetro identificativo en la petición y en la respuesta con objeto de determinar a qué solicitud corresponde cada confirmación.

La definición de los tipos de datos que aparecen en las primitivas depende de la lista de medidas, parámetros y posibles valores de ambos que se decida al modelar los Instrumentos Virtuales. Una opción es basar estos tipos en el tipo `IA5String`, como se muestra posteriormente en los ejemplos de aplicación (Capítulo 11). Además, pueden definirse como uniones los tipos, como `MEASUREMENT`, `COMVAL` y `PARVAL`, en que se considere más cómodo emplear representaciones distintas, por ejemplo, según el parámetro o la medida. En el caso de los resultados de las medidas (tipo `MEASUREMENT`), hay que considerar que también pueden ser trazas y no tratarse de un valor único, por ello una de las representaciones alternativas debe ser un registro de valores. Una posible definición de este tipo, empleando tipos de datos de ASN.1, se muestra en la Figura 6.7.

El Subsistema de Pruebas también debe comunicarse con otros dos elementos externos, la Unidad de Señalización y el EUT. La comunicación con la primera consiste en solicitar que sitúe al EUT en un determinado estado. La dificultad estriba en la construcción de la Unidad de Señalización, no en el modelado de esta interfaz, pues es tan simple como indicar la Prueba que se va a realizar. La comunicación con el EUT se realiza de forma automática o manual a través de un operario dependiendo del Método de Pruebas elegido. Se puede reservar un identificador especial de Instrumento Virtual para que represente al EUT en toda ocasión, por ejemplo, “EUT”.

```
MEASUREMENT ::= CHOICE
{
   val    IA5String,
   traza  Trace
}

Trace ::= SEQUENCE OF PairOfValues

PairOfValues ::= SEQUENCE
{
   x   REAL,
   y   REAL
}
```

Figura 6.7: Definición en ASN.1 del tipo de datos `MEASUREMENT`.

6.5 Módulo de Acceso a la Instrumentación

El Módulo de Acceso a la Instrumentación es el encargado de realizar la comunicación entre el Subsistema de Pruebas y la Instrumentación. Dado que los Juegos de Pruebas no determinan el uso de ninguna instrumentación concreta, el Módulo de Acceso a la

SPICUM servicio de publicaciones

Instrumentación debe ocultar las características específicas de la instrumentación empleada. Esto exige que se encargue de:

- Gestionar la interfaz física con la Instrumentación, adaptándola a la interfaz esperada por el Subsistema de Pruebas.
- Convertir los comandos enviados por el Subsistema de Pruebas en comandos propios de la Instrumentación y representarlos en el formato adecuado.
- Ocultar diferencias entre instrumentos de diferentes suministradores.

La estructura del Módulo de Acceso a la Instrumentación se muestra en la Figura 6.4. Se ha mantenido la misma estructura que en el Módulo de Protocolos para poder emplear las mismas herramientas en su diseño. El Modelo del Acceso a la Instrumentación contiene el código necesario para realizar la funcionalidad de este Módulo. Su comportamiento es sencillo, pues no requiere concurrencia y se limita a recibir un mensaje por la frontera superior, codificarlo, enviar uno o más mensajes por la frontera inferior, esperar la respuesta, codificarla y enviarla por donde recibió la petición original. Esto hace que el uso de SDL, como en el caso de los Sistemas de Pruebas de protocolos, para su diseño sea discutible, pues se trata más de un módulo codificador que de una máquina de estados.

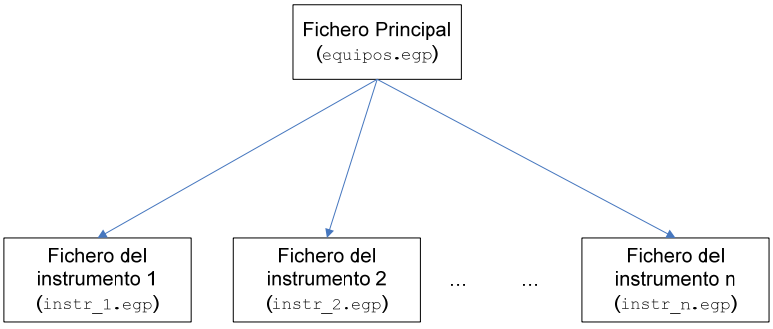


Figura 6.8: Relación entre los tipos de ficheros de configuración.

6.5.1 Implementación

El diseño realizado permite integrar en el Sistema de Pruebas instrumentación de diferentes suministradores, incluso aunque empleen un subconjunto de comandos del estándar o comandos propietarios; también tiene en cuenta el hecho de que un mismo comando tenga significados diferentes para cada equipo. El mecanismo de configuración ofrece una gran flexibilidad al Sistema de Pruebas, ya que permite utilizar instrumentación diferente con unos retoques mínimos. El acceso a la Instrumentación se ha realizado a través del bus GPIB.

La adaptación del Módulo de Acceso a la Instrumentación para que utilice unos instrumentos concretos se realiza a través de dos tipos de ficheros de configuración (Figura 6.8). El primero de estos ficheros (Figura 6.9) indica los instrumentos disponibles y la información necesaria para su direccionamiento[8]. Cada instrumento

[8] Esto incluye la dirección de la tarjeta controladora GPIB, la dirección GPIB de cada instrumento y el modo de señalización del fin de las transmisiones por el bus GPIB

SPICUM servicio de publicaciones

dispone de un fichero adicional (Figura 6.10) donde se asocian los comandos, y sus parámetros, empleados en los Juegos de Pruebas con los comandos, y sus parámetros, correspondientes del instrumento. Cada línea define la asociación de un comando; aquellos comandos que requieren una respuesta por parte del instrumento (peticiones) terminan con el carácter '?'. Los parámetros de cada comando aparecen en las líneas consecutivas a la del comando y se diferencian porque empiezan con el carácter '_'. El tipo de datos `INSTRUMENT` (Tabla 6.2-b) almacena en memoria la información de configuración de cada instrumento: direccionamiento y asociaciones de comandos.

Tabla 6.2: (a) Declaraciones de las funciones empleadas en el Modelo de Acceso a la Instrumentación y (b) Definición del tipo de datos `INSTRUMENT`.

Declaraciones	Tipos de Datos
`int InitInstrument (INSTRUMENT *instr, char *cad_error);`	`typedef struct { char id [L_ID_INSTR]; int gpib_board; short dir_gpib; int eot_mode; COMMANDS *coms; void *sig; } INSTRUMENT;`
`int ConnectReq (INSTRUMENT *instr1, PORT *port1, INSTRUMENT *instr2, PORT *port2, char *cad_error);`	
`int SetParameter (INSTRUMENT *instr, char *com, char *par, char *cad_error);`	
`int GetParameter (INSTRUMENT *instr, char *com, char *par, char *valor, int len, char *cad_error);`	
`int StopInstrument (INSTRUMENT *instr, char *cad_error);`	
(a)	*(b)*

El Modelo de Acceso a la Instrumentación se ha escrito en C y consiste en un conjunto de cinco funciones que procesan las distintas primitivas empleadas por el Subsistema de Pruebas; estas funciones son independientes de la interfaz física ofrecida por la Instrumentación y ofrecen una API genérica que puede ser empleada en cualquier Sistema de Pruebas radio. La Tabla 6.2-a muestra las declaraciones de las funciones que se han implementado. Todas estas funciones devuelven un valor distinto de cero si se ha producido un error (el código del error) y en el último parámetro (`cad_error`) una breve explicación del mismo.

```
# Configuration file for Test System
#id_equipo  GPIB_board   dir_GPIB   EOT_mode    file
Spec_An         0           20         2       specan.egp
Wave_Gen        0           29         2       wavgen.egp
```

Figura 6.9: Ejemplo de fichero principal de configuración.

La explicación de estas funciones es equivalente a la de las primitivas descritas en la Sección 6.4.2. Por ejemplo, la función `GetParameter` toma como primer parámetro el instrumento al que se le solicita que realice la medida; los dos siguientes parámetros identifican la medida a realizar; los parámetros cuarto y quinto devuelven el resultado de la medida y la longitud en caracteres de la misma, respectivamente; el último parámetro sólo se utiliza en caso de error. La carga de la información de configuración

SPICUM servicio de publicaciones

de cada instrumento se realiza durante su inicialización (función `InitInstrument`). Cuando se recibe un mensaje, se comprueba su tipo y se codifican los correspondientes comandos de los instrumentos. En caso de que un comando de Prueba equivalga a más de un comando de la Instrumentación, estos se ejecutan secuencialmente. Cuando todos los comandos han sido ejecutados, se devuelve una respuesta confirmando la operación. El bus GPIB de un instrumento puede encontrarse en dos estados: local y remoto. El estado local se ha considerado el estado de reposo.

```
# DEVICE: Spectrum Analyzer (specan.egp)

# Test command      Device command

# Non-query commands

Reset               *RST
Span                SENS1:FREQ:SPAN
Center              SENS1:FREQ:CENT

Trigger             TRIG1:SEQ:SOUR
_FreeRun            IMM
_Line               LINE
_RFPower            RFP

# Query commands

PeakPower?          CALC1:MARK1:Y?
RefLevel?           DISP:WIND1:TRAC1:Y:SCAL:RLEV?
SweepTime?          SENS1:SWE:TIME?

Detector?           SENS1:DET1:FUNC?
_Average            AVER
_Sample             SAMP
_Rms                RMS
```

Figura 6.10: Ejemplos de asociación de comandos en el fichero de configuración para el analizador de espectros FSIQ26.

El codificador Codec $Local_{PHY}$ se ha desarrollado para instrumentación que utilice el bus GPIB y comandos SCPI para su control remoto (ver Sección 6.5.1.1), ya que es la solución más extendida en el ámbito de la instrumentación. Por su parte, el codificador de la interfaz con el Subsistema de Pruebas (Codec $Local_{TTCN}$) es el mismo que el empleado en los Sistemas de Pruebas de protocolos; se emplea la sintaxis de transferencia ASCII.

6.5.1.1 Bus GPIB

El bus GPIB (*General Purpose Interface Bus*) [IEEE 488] es un bus asíncrono utilizado de forma habitual para el control remoto de instrumentación con velocidades de transferencia de hasta 8 MB/s. Uno de los dispositivos actúa en el papel de controlador del bus. La comunicación se realiza mediante comandos, que son enviados y recibidos a través de las funciones listadas en la Tabla 6.3 [IEEE 488-2].

Los comandos utilizados para el control de la instrumentación están definidos en el estándar SCPI (*Standard Commands for Programmable Instrumentation*) [SCPI99][9]. Los

[9] Se trata de una evolución de los comandos especificados en la parte 2 del estándar GPIB

SPICUM servicio de publicaciones

comandos están basados en una estructura jerárquica. Un comando se forma mediante la concatenación de los nombres de todos los nodos que atraviesa hasta llegar a la hoja que representa el comando a ejecutar; como separador se utiliza el carácter ':'. Los comandos se transmiten como cadenas ASCII. Por ejemplo, el comando, tercer parámetro (`Data_Buffer`) de la función `Send`, que ajusta la frecuencia central de un analizador de espectros tiene la forma:

```
"SENSe:FREQuency:CENTer <valor_numérico>"
```

Tabla 6.3: Funciones de alto nivel para el acceso al bus GPIB.

Función	Descripción
`void Send (int Board_Index, short Device_Address, void *Data_Buffer, long Byte_Count, int EOT_Mode)`	Envía un mensaje a un instrumento.
`void Receive (int Board_Index, short Device_Address, void *Data_Buffer, long Byte_Count, int Termination)`	Recibe un mensaje de un instrumento.
`void SendIFC (int Board_Index)`	Inicialización. Fuerza a una placa GPIB a ser la controladora maestra del bus, y asegura que todos los dispositivos conectados están en reposo.
`void EnableRemote (int Board_Index, short Address_List[])`	Inicialización. Habilita la programación remota de un instrumento vía el bus GPIB.
`void EnableLocal (int Board_Index, short Address_List[])`	Permite el control local (panel frontal) de los instrumentos indicados.

6.6 Conclusiones

En este capítulo se ha mostrado que es posible diseñar Sistemas de Pruebas radio utilizando la misma arquitectura que para los Sistemas de Pruebas de protocolos. Esta asimilación permite emplear todas las herramientas ya existentes en el área de las pruebas de protocolos y construir elementos comunes para ambos tipos de pruebas, tales como el control de la ejecución, la generación de informes, el registro de la prueba, etc., redundando en una disminución de los costes de desarrollo. Esta arquitectura permite la integración de instrumentos de diferentes fabricantes así como la inmediata sustitución de cualquier equipo por otro con capacidades similares.

El uso de la notación TTCN para modelar los Casos de Prueba radio permite incrementar su grado de formalización, evitando las ambigüedades inherentes al lenguaje natural utilizado actualmente para la definición de las Pruebas radio. Esto simplificaría el proceso de validación de los Sistemas de Pruebas, ya que el comportamiento de las mismas habría sido acordado previamente por los organismos de estandarización. Se ha propuesto una interfaz de primitivas y comandos abstractos para el control de la Instrumentación desde el Caso de Prueba. Esta interfaz abstracta se puede particularizar para cualquier equipo mediante el uso de ficheros que conviertan la orden abstracta en la secuencia de comandos requerida por la Instrumentación. En el Capítulo 11 se presentan ejemplos de aplicación de esta metodología de diseño de Sistemas de Pruebas radio para las tecnologías Bluetooth y UMTS.

SPICUM servicio de publicaciones

SPICUM
servicio de publicaciones

Capítulo

7

"Mirad el dibujo de esta concha marina. La espiral moteada se curva hacia el interior hasta el infinito. Ésta es la estructura del universo. Hay una presión constante que empuja hacia dentro, una tendencia de la materia a evolucionar hacia formas cada vez más complejas."

Hiroko, Marte Verde

CAPÍTULO 7: DISEÑO DE SISTEMAS CON SDL

En la Metodología de Diseño descrita en el Capítulo 4, se sugirieron unas líneas generales para la construcción del Módulo de Protocolos, aunque se indicó que no se iba a optar por una metodología específica. Este Módulo de Protocolos se modela en SDL, lenguaje que utiliza numerosos conceptos y estructuras que siguen la filosofía de la orientación a objetos.

Existe un amplio abanico de metodologías generalistas pensadas para el diseño basado en orientación a objetos [BURK96]. Entre ellas se pueden citar las metodologías de Booch ([BOOC94], [BOOC95]), Coad/Yourdon [COAD91], Coleman [COLE94], Jacobson [JACO92] y Rumbaugh [RUMB91]. Esta última, denominada OMT (*Object Modeling Technique*) es una de las más utilizadas.

Para diseños basados en el lenguaje SDL se han propuesto metodologías más específicas. Entre ellas se encuentran las siguientes:

- El proyecto SPECS (*Specification and Programming Environment for Communication Software*) ([SPEC93], [OLSE94]) definió una metodología general para el uso de lenguajes de especificación formal en la ingeniería. Esta metodología fue adaptada para el uso de SDL, ofreciendo un conjunto de directrices para la creación de modelos SDL. Esta metodología identifica cinco niveles (Estructura, Comportamiento, Datos, Tipos de Componentes y Ubicación de Tipos) en el proceso de diseño, cada uno de los cuales consta de varios pasos.
- La metodología SDL+ [REED96] presupone que para la mayoría de los productos la eficiencia en ejecución no es un aspecto primordial, pero sí lo son los costes del software y la necesidad de cumplir una ventana temporal de puesta del producto en el mercado. Según esto, propone cinco pasos: documentación,

SPICUM servicio de publicaciones

análisis, bosquejo del diseño, formalización (siguiendo las directrices de [SPEC93]) e implementación.

- La metodología SOMT (*SDL-oriented Object Modeling Technique*) [EKAN95] es una adaptación de la metodología OMT, aunque también recoge influencias de otros métodos como el de Jacobson y el de Booch. Define cinco actividades, dentro de cada cual se construyen diversos modelos que permiten ir acercándose al objetivo final.
- [BORN98] describe una metodología de diseño para sistemas distribuidos que utiliza modelos SDL para identificar posibles errores en la especificación del sistema antes de proceder a su implementación. Propone partir de una especificación en ODL para generar, automáticamente, un modelo estructural en SDL de la misma, al cual se le va añadiendo y refinando el comportamiento[1].
- En [VERI96] se describe cómo usar eficientemente las técnicas de orientación a objetos en sistemas de tiempo real mediante una combinación de OMT, SDL y MSC. El lenguaje SDL se recomienda para el diseño de la arquitectura y el diseño detallado, así como la implementación; el lenguaje MSC lo utiliza para el análisis de requisitos y la producción de Casos de Prueba.
- La metodología TIMe [BRÆK99] recomienda el uso de una combinación de lenguajes y notaciones para el análisis, modelado e implementación de sistemas reactivos, concurrentes, de tiempo real, distribuidos, heterogéneos y complejos. Entre estos lenguajes se encuentran SDL, UML y MSC. El proceso de desarrollo se divide en las etapas de análisis, diseño, implementación e instanciación. El uso de SDL está recomendado para la fase de diseño, donde se modelan la estructura y el comportamiento del sistema.
- Otros usos de SDL como parte de una metodología de diseño incluyen: a) especificación de requisitos para servicios de telecomunicación (RATS – *Requirements Assistant for Telecommunications Services*) [ARMI97]; b) dentro de un proceso de diseño en V, tanto para un prototipado rápido como para el desarrollo final [BADR07]; d) reutilización de código a través del uso de patrones ([BUSC96], [GEPP01]).

En este Capítulo se describe el análisis realizado de la metodología SOMT. Para esta evaluación se ha implementado el Nivel de Red de DECT (el Anexo F contiene una breve descripción de sus especificaciones) siguiendo los pasos definidos en esta metodología. La metodología elabora distintos modelos donde se va refinando la descripción del sistema; emplea modelos de objetos, diagramas MSC y SDL y documentos textuales. Los elementos de un modelo se enlazan con sus precursores en los modelos previos, lo que permite trazar un elemento hasta su concepción inicial.

Como resultado de este estudio se han propuesto modificaciones de la metodología para su adaptación a procesos de diseño similares al analizado; estas modificaciones han afectado fundamentalmente al tipo de modelos empleados en cada fase. La metodología resultante, que se ha denominado M-SOMT (*Modified SOMT*) ([ALBA00a], [ALBA00b], [ALBA99]), posee las ventajas del enfoque orientado a objetos, así como las derivadas del uso de lenguajes formales, pero está adaptada a las características de un proceso de diseño basado en un estándar.

[1] De forma similar, en [WITA95] se presenta un conjunto de reglas para transformar una especificación OMT en un modelo SDL'92, el cual es refinado posteriormente.

SPICUM servicio de publicaciones

En primer lugar, se explican las actividades y los modelos que propone la metodología SOMT. A continuación, se describen las tareas realizadas en cada una de las fases propuestas por la metodología, junto con los modelos generados en cada una de ellas, para el diseño del Nivel de Red de DECT. Esta descripción está enfocada a la metodología, por lo que los modelos en sí no se describen en detalle. Seguidamente, se resume la propuesta de metodología modificada, M-SOMT. El modelo construido del Nivel de Red se describe en el Anexo K.

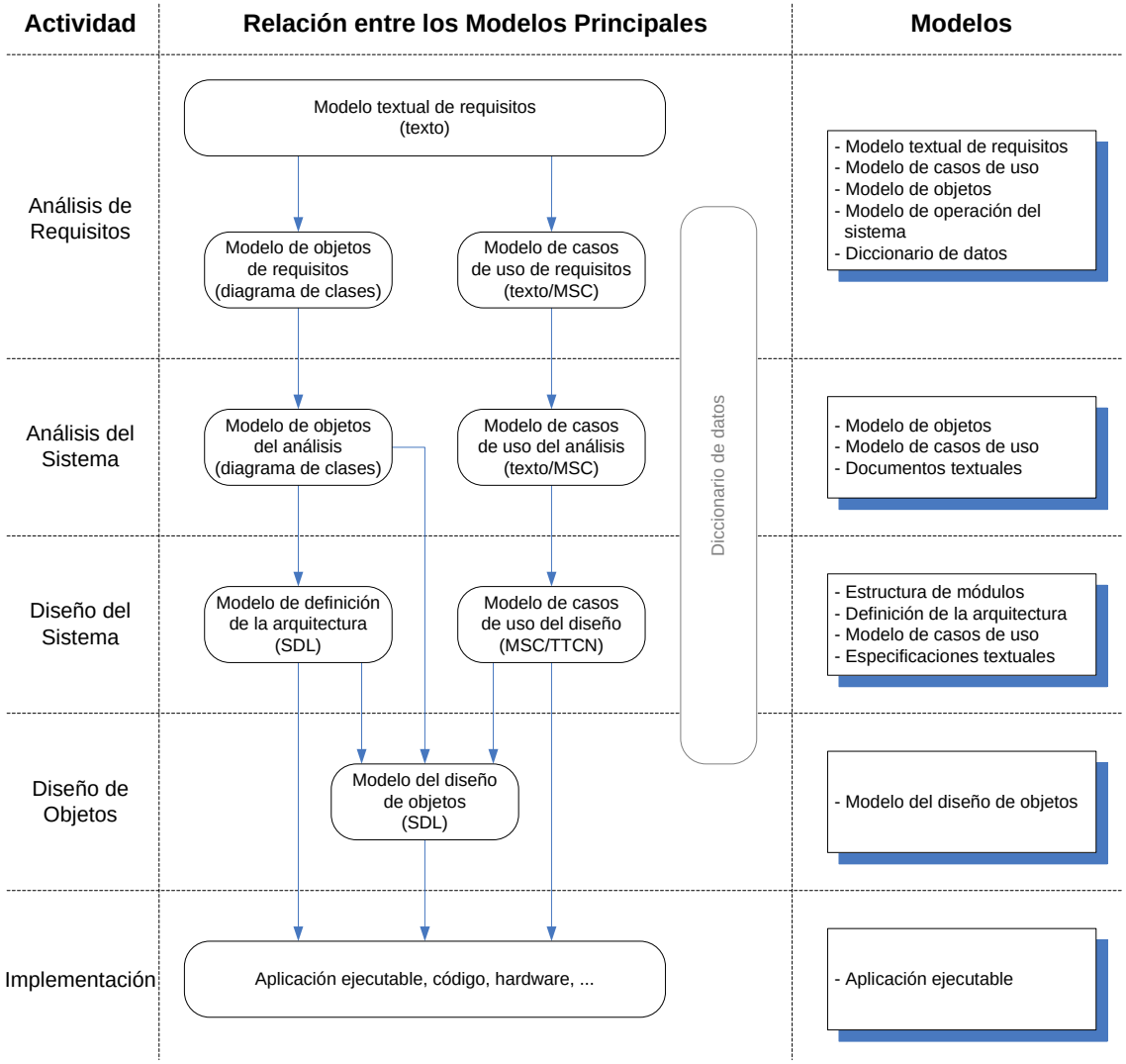


Figura 7.1: Actividades y modelos de la metodología SOMT.

7.1 Visión General de la Metodología SOMT

La metodología SOMT (*SDL-oriented Object Modeling Technique*), propuesta por Anders Ek [EKAN95], integra el análisis orientado a objetos con el diseño formal con SDL. Esta metodología sugiere un conjunto de actividades o fases, y un orden entre ellas, para realizar el diseño e implementación de sistemas bajo el paradigma de la orientación a objetos adaptado al lenguaje SDL. Se trata de una adaptación de la metodología OMT [RUMB91], cuya notación utiliza, aunque también recoge influencias

SPICUM servicio de publicaciones

de UML ([BOOC94], [BOOC95]) y de otros métodos de análisis como el de Jacobson [JACO92], por ejemplo, la idea de los modelos de casos de uso. Ejemplos de uso de la metodología SOMT son [KEUM98], [YEWE00] y [RODR07].

Cada una de las actividades (Figura 7.1) de la metodología se encarga de uno o más aspectos del proceso de diseño; el resultado de cada actividad es un conjunto de modelos, uno o más, que serán empleados en actividades posteriores. La metodología define las siguientes cinco actividades:

- Análisis de requisitos
- Análisis del sistema
- Diseño del sistema
- Diseño de objeto
- Implementación

Aunque en la metodología SOMT la descripción de las diferentes actividades se hace de forma secuencial, esto no implica que en la práctica estas actividades tengan que realizarse secuencialmente. Por el contrario, dependiendo del tamaño y complejidad de la aplicación, las distintas actividades se organizarán de una u otra forma. El enfoque de la metodología es iterativo, no secuencial. La progresión entre diferentes modelos es un proceso creativo que supone la toma de decisiones de ingeniería. Se trata de una actividad manual, aunque asistida por las herramientas.

Una parte de la metodología son directrices para llevar a cabo la transición entre los diferentes modelos. En muchos casos, la metodología establece relaciones entre los componentes de varios de estos modelos. Un tipo de relación especialmente importante es la denominada *implink* (*implementation link*). Esta relación se utiliza cuando un componente de un modelo se puede ver como la implementación de otro[2].

A continuación se describen cada una de las actividades y los modelos que se generan.

7.1.1 Actividades

La Figura 7.1 muestra las actividades que forman la metodología (izquierda), junto con los modelos que se generan en cada una de ellas (derecha). Los modelos principales de cada actividad y las relaciones entre ellos se muestran en la parte central de la figura.

La primera actividad de la metodología es el análisis de requisitos. Su objetivo es analizar el dominio del problema y obtener los requisitos del sistema que se desea construir. Para ello, el sistema se considera como una caja negra que interacciona con su entorno. En esta fase se utilizan cinco modelos principales:

- Modelo textual de requisitos: Es una especificación convencional de requisitos expresada por escrito.
- Modelo de casos de uso de requisitos: Trata de capturar y validar los requisitos desde el punto de vista del usuario para asegurar que el sistema resolverá el

[2] Las herramientas pueden soportar la evolución del diseño copiando un elemento de un modelo en un modelo posterior, adaptándolo automáticamente a las características del este último. Por ejemplo, una clase del modelo de objetos se puede pegar como un proceso de SDL en el modelo de diseño del sistema.

SPICUM servicio de publicaciones

problema correcto. Consta de un conjunto de patrones de utilización, cada uno de los cuales se describe usando texto estructurado o diagramas MSC y HMSC[3].

- Modelo de objetos de requisitos: Persigue un doble objetivo. Por un lado, describir mediante diagramas de contexto el sistema y los actores externos que interaccionan con él. Por otro, documentar los conceptos utilizados durante el análisis de requisitos y las relaciones entre ellos. Incluye uno o varios diagramas que ilustran un conjunto de objetos y su relación, incluyendo relaciones de herencia y agregación; también puede incluir diagramas de estados.
- Modelo de operación del sistema: Describe las acciones atómicas que el sistema debe ser capaz de realizar.
- Diccionario de datos: Lista los conceptos identificados en esta fase, junto con breves explicaciones de los mismos.

La segunda actividad es el análisis del sistema, que busca identificar la arquitectura y los objetos principales que debe contener para que proporcione la funcionalidad esperada. Los modelos utilizados en esta actividad son:

- Modelo de objetos del análisis: Describe la arquitectura lógica del sistema e identifica sus objetos. Incluye diagramas de clases y, posiblemente, diagramas de estados. A diferencia del modelo de igual nombre empleado en la actividad de análisis de requisitos, se centra en la estructura de objetos interna del sistema y en describir el propio sistema. Para cada elemento, incorpora qué información mantiene (atributos), las acciones que debe realizar (operaciones), qué otros objetos necesita (relaciones de asociación y composición) y las analogías con otros objetos (relaciones de herencia).
- Modelo de casos de uso del análisis: Representa el aspecto dinámico del sistema, mostrando las interacciones de algunos de los objetos. La metodología propone la utilización de dos tipos de casos de uso: refinamientos del modelo de la fase de análisis de requisitos, que denominaremos casos de uso propiamente dichos y que derivan del modelo de casos de uso de la actividad anterior, y patrones de comportamiento.
- Documentos textuales: Recogen las decisiones y requisitos relacionados con la arquitectura que no se han podido expresar en el modelo de objetos. Sirven de apoyo al resto de modelos elaborados en esta actividad.

La actividad de diseño del sistema se emplea para definir con precisión la arquitectura del sistema, incluyendo las interfaces entre las distintas partes que lo componen. También se hacen consideraciones sobre estrategias de implementación y descomposición del trabajo en tareas con vistas a su asignación a distintos grupos de trabajo. Los modelos que produce esta actividad son los siguientes:

- Estructura de módulos del diseño: Describe los módulos que componen el diseño. Se modelará en SDL.
- Definición de la arquitectura: Incluye diagramas SDL con la arquitectura de bloques y la definición de las interfaces entre los componentes del sistema mediante señales o procedimientos remotos. Establece una descomposición lógica de la funcionalidad del sistema.

[3] HMSC (*High-Level Message Sequence Chart*) es una extensión de la notación MSC que permite definir gráficamente cómo se combinan los comportamientos de un conjunto de MSCs.

SPICUM servicio de publicaciones

- Modelo de casos de uso del diseño: Utiliza diagramas MSC y HMSC para definir las interfaces entre los componentes del sistema. Deben ofrecer un nivel de detalle suficiente para ser utilizados como especificaciones de prueba.
- Especificaciones textuales del diseño: Como en la actividad de análisis del sistema, se encargan de recoger aquellos aspectos de los componentes que no se pueden formalizar. Su objetivo es completar la visión estática y dinámica ofrecida por los demás modelos de esta actividad.

La actividad de diseño de objetos crea una descripción completa y verificada del comportamiento del sistema, para lo que utiliza, esencialmente, diagramas SDL. Es el único modelo, denominado modelo del diseño de objetos, que genera esta actividad. La construcción de los diagramas se puede descomponer en tres pasos:

- Representar los conceptos del modelo de objetos del análisis del sistema. Hay que considerar si cada objeto es un elemento activo, representado mediante procesos, o pasivo, representado mediante tipos de datos, y su ubicación en el paquete o sistema apropiado.
- Elegir un conjunto de casos de uso básicos y definir los elementos SDL que los implementan, considerando sólo el comportamiento normal
- Elaborar el diseño mediante la introducción de nuevos casos de uso y el refinamiento de la descripción SDL, incluyendo las situaciones excepcionales, como el manejo de errores. Se trata de un proceso iterativo. Cada iteración requiere la prueba y verificación de la funcionalidad ya implementada.

La quinta y última actividad es la implementación. Esta etapa genera una aplicación ejecutable que satisface los requisitos. Las tareas de esta fase dependen en gran medida del entorno de uso de la aplicación, pero en cualquier caso deben incluir: el uso de una herramienta de generación automática de código a partir de un sistema SDL; la adaptación del código generado al entorno de operación; la integración del código en los requisitos hardware; y la ejecución en el entorno final de las pruebas definidas a partir de los casos de uso identificados en las actividades anteriores.

En las siguientes secciones se describe la aplicación de cada una de las actividades de la metodología SOMT al diseño del Nivel de Red del sistema DECT.

7.2 Análisis de Requisitos

El objetivo de esta actividad es analizar el dominio del problema y los requisitos impuestos por el usuario. Sin embargo, en este caso el diseño está basado en un conjunto de estándares ([ETS 300 175], [ETS 300 444]) durante cuya elaboración se habrán realizado gran parte de las tareas habituales de esta actividad. El primero de ellos, [ETS 300 175], describe la Interfaz Común, arquitectura y funcionalidad para servicios de voz y datos, del sistema DECT; el segundo, [ETS 300 444], describe el perfil para servicios de telefonía de voz. La información proporcionada por los modelos resultantes de esta actividad se encuentra ya recogida, en su mayoría, en estos estándares. Por ello, como se justifica posteriormente, puede obviarse su elaboración.

Tanto el modelo textual de requisitos como el modelo de objetos de requisitos contienen información ya recogida en los estándares indicados pues proporcionan, respectivamente, una descripción exacta, completa y estructurada de cómo debe ser el sistema y una representación de los objetos que interaccionan con el sistema y las

SPICUM servicio de publicaciones

correspondientes interacciones. Lo mismo ocurre con el modelo de casos de uso y el modelo de operación del sistema, aunque su descripción en los estándares es textual y no mediante diagramas MSC/HMSC. Como ejemplos, la Figura 7.2 muestra la explicación textual correspondiente a la identificación de la Terminación Portátil, y la Figura 7.3 representa los posibles estados de los enlaces DLC y las transiciones entre ellos.

Cuando recibe la primitiva MM_IDENTITY-req, FT inicia el procedimiento de identificación enviando un mensaje {IDENTITY-REQUEST} a PT a y activa el temporizador <MM_ident.2>. Este mensaje especifica el tipo de identidad requerido mediante el elemento de información <<IDENTITY-TYPE>>.

Al recibir el mensaje {IDENTITY-REQUEST}, PT entrega la primitiva MM_IDENTITY-ind. Cuando recibe la primitiva MM_IDENTITY-res, PT envía el mensaje {IDENTITY-REPLY} que contiene los parámetros de identificación solicitados por FT. Si no se puede proporcionar el parámetro solicitado, el mensaje {IDENTITY-REPLY} no contendrá información de identificación. Este mensaje actúa como rechazo de la identificación.

Al recibir el mensaje {IDENTITY-REPLY}, FT detiene el temporizador <MM_ident.2> y entrega la primitiva MM_IDENTITY-cfm.

Este procedimiento está controlado por el temporizador <MM_ident.2> en FT. El primer vencimiento de este temporizador provoca la retransmisión del mensaje {IDENTITY-REQUEST}. Si vence una segunda vez, FT termina el procedimiento y libera la transacción.

Figura 7.2: Procedimiento de identificación de la Terminación Portátil (PT).

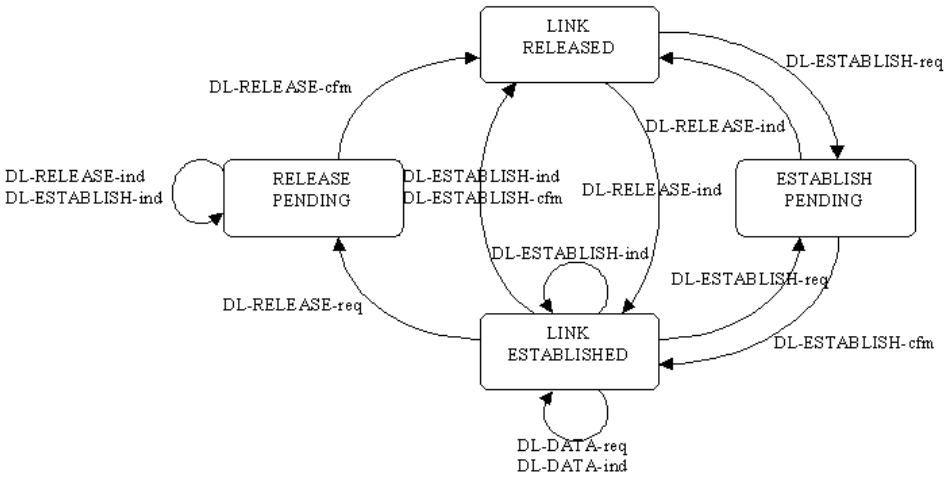


Figura 7.3: Estados de los enlaces DLC vistos desde la entidad LCE del Nivel de Red.

Por tanto, la única tarea por realizar en esta actividad es la elaboración del diccionario de datos. El diccionario de datos incluye los conceptos manejados en esta actividad junto con explicaciones de los mismos. Aunque la información de este modelo se encuentra recogida en el estándar[4], se ha considerado conveniente adaptar su

[4] La norma [ETS 300 175-1] recoge gran parte de estas definiciones.

SPICUM servicio de publicaciones

presentación (y formato) a nuestras necesidades por dos razones. En primer lugar, el objetivo del diccionario de datos difiere del propósito con que fueron creados los capítulos de definiciones incluidos en el estándar. Estos últimos no contienen una definición completa del sistema ya que sólo buscan ser un apoyo al resto de capítulos de la norma. Sin embargo, el diccionario de datos pretende ser un modelo independiente que contenga una descripción completa de los aspectos básicos del sistema. En segundo lugar, el diccionario de datos constituye el primero de los modelos que se elaboran en el marco de la herramienta. Esta integración hace que este modelo sea el punto de partida ideal para los enlaces de implementación (*implinks*)[5].

Establecimiento iniciado por el CC de FT - Agrupa al conjunto de procedimientos relacionados con el establecimiento de la llamada entrante, iniciado por la entidad CC correspondiente al FT. (Procedimiento de CC)

Establecimiento iniciado por el CC de PT - Agrupa al conjunto de procedimientos relacionados con el establecimiento de la llamada saliente, iniciado por la entidad CC correspondiente al PT. (Procedimiento de CC)

Identificación de PT - Procedimiento utilizado por FT para solicitar a PT su identificación mediante parámetros específicos como IPUI o IPEI. (Procedimiento de MM)

Liberación anormal de LCE - Procedimiento utilizado para llevar a cabo la liberación inmediata e incondicional del enlace descartando los mensajes encolados sin notificación. (Procedimiento de LCE)

Liberación normal de LCE - Procedimiento utilizado para llevar a cabo la liberación condicional del enlace permitiendo a DLC la transmisión de los mensajes encolados. (Procedimiento de LCE)

Figura 7.4: Conceptos de la sección de Acciones del diccionario de datos.

El diccionario de datos incluye las secciones de Nombres[6], con conceptos que van desde entidades del Nivel de Red hasta estados de las mismas o mensajes, y Acciones[7], donde hay una entrada para cada procedimiento realizado por cada una de las entidades (Figura 7.4). La sección de Relaciones[8] sugerida en SOMT no aparece de forma explícita, aunque estas relaciones estáticas están presentes en las definiciones. Las definiciones de los conceptos básicos se han extraído de la norma [ETS 300 175-1], las definiciones de las identidades han salido de la norma [ETS 300 175-6] y las definiciones de conceptos propios del Nivel de Red se han elaborado, fundamentalmente, a partir de la norma [ETS 300 175-5].

La vista que aparece en la herramienta tras la finalización de esta actividad se muestra en la Figura 7.5. Hay una sección para cada actividad, y el único documento incluido hasta el momento es el diccionario de datos. Los estándares relacionados no se han incorporado a la herramienta porque no ofrece esta posibilidad. Esto dificultará la

[5] Estos enlaces son, como se ha indicado anteriormente, relaciones entre un concepto y su implementación.

[6] Agrupa las definiciones de los elementos presentes en el desarrollo, tanto elementos activos como elementos pasivos.

[7] Contiene breves explicaciones de los procedimientos que cada una de las entidades activas presentadas en la primera sección es capaz de realizar.

[8] Establece relaciones estáticas entre los elementos definidos en la sección anterior como puede ser la combinación de varios de estos elementos para formar otro de ellos.

SPICUM servicio de publicaciones

conexión de la información que contienen con los modelos de actividades posteriores; la relación se establecerá en términos textuales mediante una referencia a los correspondientes apartados de la norma.

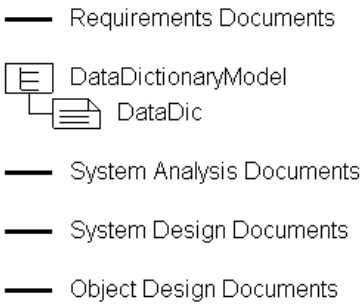


Figura 7.5: Vista del organizador de documentos tras la actividad de análisis de requisitos.

7.3 Análisis del Sistema

El objetivo de esta actividad es estudiar, con un enfoque orientado a objetos, la información aportada por la especificación de requisitos y los modelos construidos en la actividad anterior.

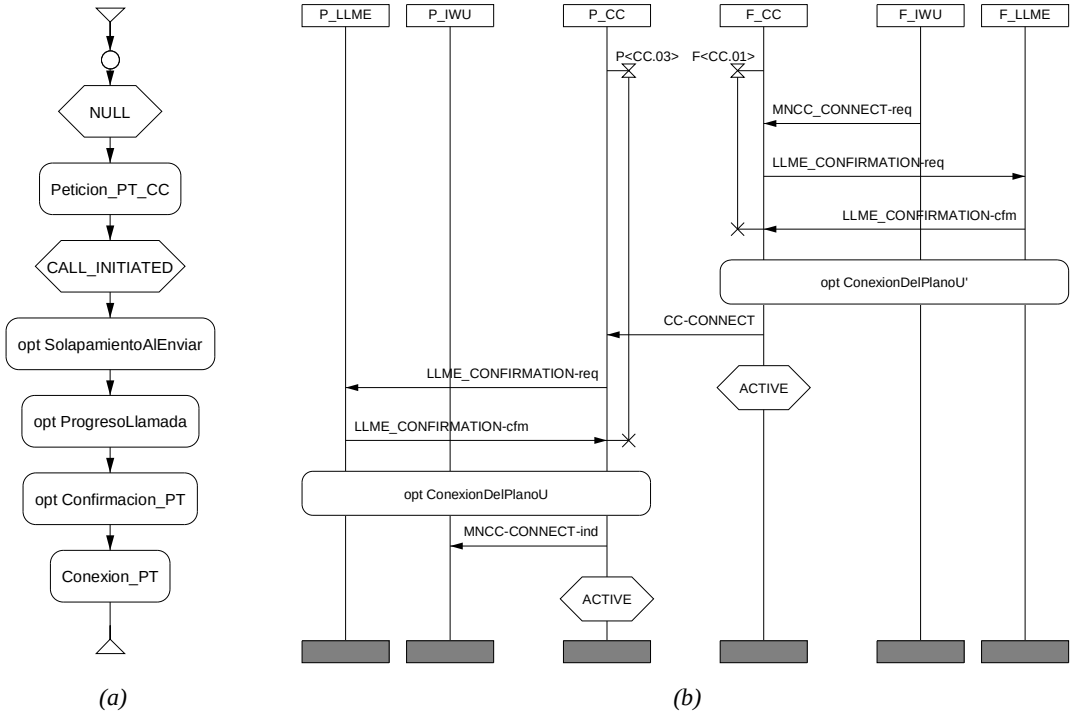


Figura 7.6: Diagramas que representan (a) el proceso de establecimiento de llamada iniciado por la Terminación Portátil (diagrama HMSC) y (b) la interacción que realiza la conexión (diagrama MSC).

En el modelo de casos de uso se ha distinguido entre los casos de uso propios, procedentes del refinamiento de los casos de uso de la actividad anterior, y los patrones de comportamiento. Todos ellos se han elaborado a partir de la información recogida en

SPICUM servicio de publicaciones

el estándar. Los procedimientos definidos en el estándar han dado lugar a los casos de uso, mientras que los patrones de comportamiento representan agrupaciones de acciones porque se repiten en varios procedimientos o porque aclaran la representación de los casos de uso. Los casos de uso se pueden ver, por tanto, como representaciones de un mayor nivel que los patrones de comportamiento. En las ocasiones en que la utilización de los patrones de comportamiento no ha sido suficiente y se ha requerido un nivel superior de organización, se han utilizado diagramas HMSC.

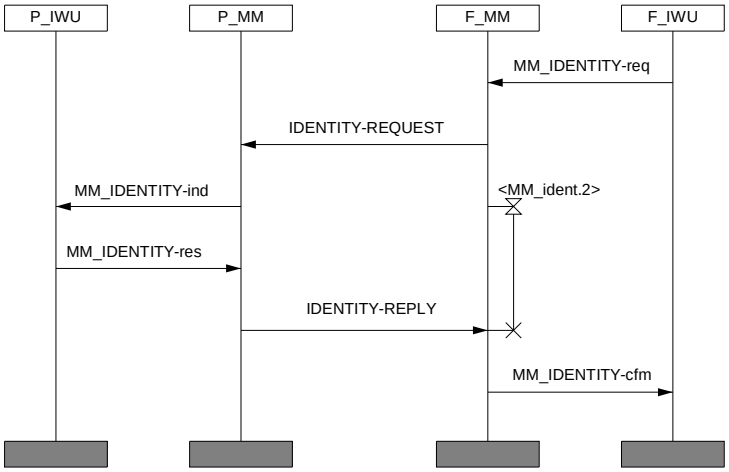


Figura 7.7: Caso de uso del procedimiento de identificación de la Terminación Portátil (PT).

Los diagramas se han agrupado según la entidad del Nivel de Red a que corresponden: Control de la Llamada (`CC` – *Call Control*), Gestión de la Movilidad (`MM` – *Mobility Management*) y Control del Enlace (`LCE` – *Link Control Entity*). Tan sólo se han utilizado diagramas HMSC para la entidad de Control de la Llamada. A modo de ejemplo, la Figura 7.6 muestra el diagrama correspondiente al establecimiento de llamada iniciado por PT, que se ha representado mediante un diagrama HMSC, y el patrón de comportamiento que modela la conexión en sí. Por su parte, la Figura 7.7 es el resultado de modelar, en esta actividad, el procedimiento descrito en la Figura 7.2.

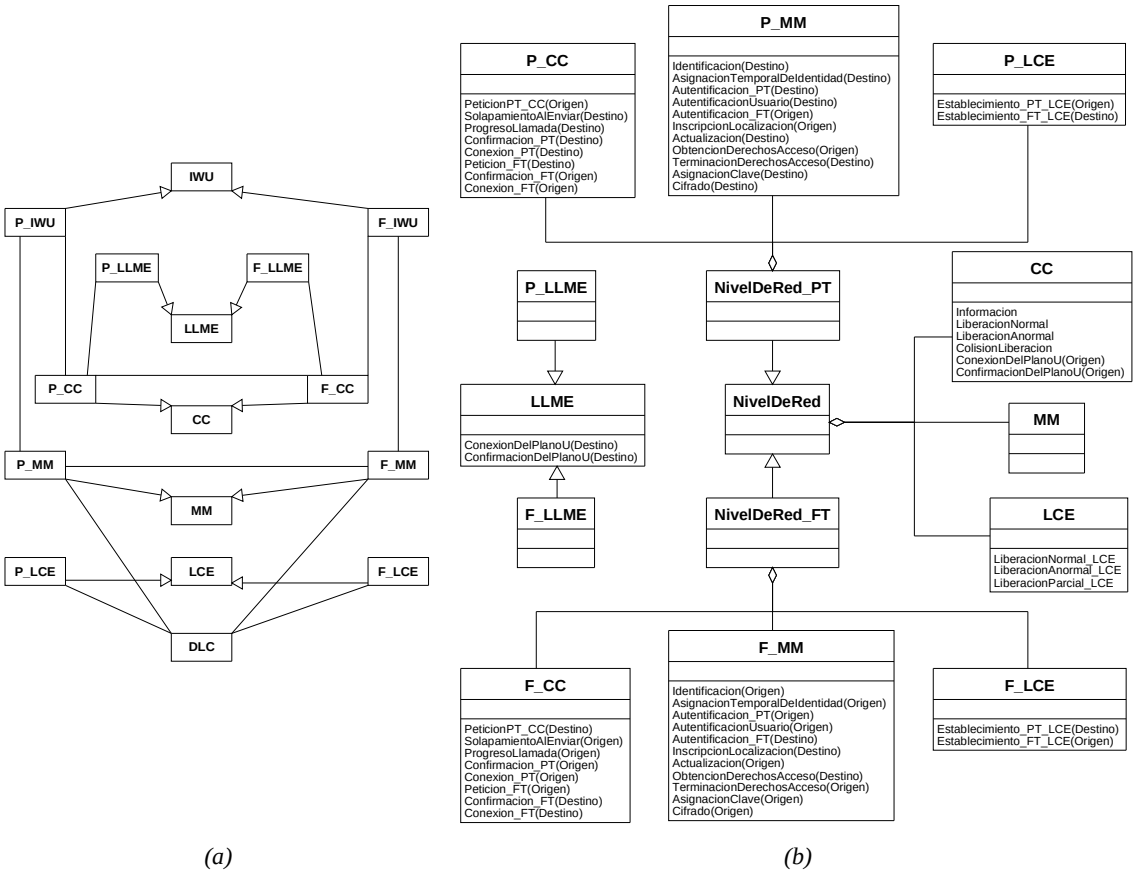
En los diagramas se prescinde de la información en cuanto a valores de los mensajes y primitivas intercambiados, modelando únicamente los envíos y recepciones de mensajes y las operaciones con los temporizadores. Además, se incluyen comentarios textuales que completan la descripción, debido a la carencia de expresividad de la notación MSC. La metodología propone que se utilice la documentación textual de esta actividad, pero se ha optado por incorporar estas aclaraciones a los propios diagramas.

El modelo de objetos del análisis representa las relaciones existentes entre las distintas entidades, ofreciendo un punto de vista estático del sistema; es, por tanto, una visión complementaria a la ofrecida en el modelo de casos de uso. Analizando el modelo de casos de uso, se identifican cinco entidades en la Terminación Fija: tres del Nivel de Red (`F_CC`, `F_MM` y `F_LCE`), una del Nivel de Gestión (`F_LLME`) y una del Nivel de Aplicación (`F_IWU`). Para la Terminación Portátil se identifican otras tantas entidades (`P_CC`, `P_MM`, `P_LCE`, `P_LLME`, `P_IWU`). Además, hay una última entidad que representa el punto de conexión entre ambos extremos de la comunicación (`DLC`).

SPICUM servicio de publicaciones

Cada una de estas entidades origina una clase. Además, dado que las entidades homólogas de las Terminaciones Fija y Portátil presentan numerosas similitudes, se ha optado por crear también unas clases genéricas (`CC`, `MM`, `LCE`, `LLME`, `IWU`) que incluyan estas características. Así, se llega al diagrama plegado del modelo de objetos (Figura 7.8-a).

El diagrama expandido representa las relaciones de asociación y composición entre estas clases (Figura 7.8-b). Se ha incluido en este diagrama una clase para el Nivel de Red, que se construye como la agregación de las correspondientes entidades (`CC`, `MM` y `LLME`). Este diagrama contiene también algunas relaciones de herencia, allí donde no ofusca la representación.



(a) *(b)*

Figura 7.8: Diagrama (a) plegado y (b) expandido del modelo de objetos del análisis.

Las operaciones de cada entidad se obtienen a partir del modelo de casos de uso. Básicamente se requieren dos operaciones para cada caso de uso, una correspondiente al extremo origen y otra al extremo destino. Las operaciones comunes a ambos extremos se han asociado a la clase genérica. La Figura 7.8-b muestra las operaciones de cada una de las clases. Para las entidades ajenas el Nivel de Red, al ser objetos externos al sistema, no se ha considerado qué operaciones deben realizar. Cada clase que aparece en los diagramas del modelo de objetos se ha conectado a las correspondientes definiciones del diccionario de datos mediante los *implinks*.

A la hora de elaborar los documentos textuales de esta actividad, nos hemos apartado del objetivo propuesto por la metodología, ya que, como las normas describen todos los

SPICUM servicio de publicaciones

aspectos del sistema, no supone un grave inconveniente que se queden aspectos sin expresar en los modelos elaborados en esta actividad. Por ejemplo, no hace falta elaborar un documento auxiliar con los valores que transporta cada mensaje intercambiado en los procedimientos. Por el contrario, se ha hecho uso de esta documentación para aclarar la notación empleada en los diagramas del modelo de casos de uso, ya que se han empleado extensiones de la norma [Z.120] (Tabla 7.1). Esto hace que esta documentación no pueda entenderse como un modelo.

Tabla 7.1: Extensiones adoptadas para los diagramas con notación MSC.

Concepto	Problema	Ejemplo	Solución adoptada
Acciones asociadas al vencimiento de un temporizador	Identificar cuándo se produce el vencimiento para colocar la llamada al diagrama correspondiente	Liberación normal de llamada: Temporizador `<CC.02>`	Elaborar un patrón de comportamiento, pero no invocarlo en el diagrama del caso de uso sino indicarlo textualmente. Nombre genérico: `VencimientoNombreTemporizador` (ej: `VencimientoCC02`).
Reinicio de un temporizador	Como respuesta a un envío, se puede recibir un mensaje solicitando el reinicio del temporizador iniciado	Petición de establecimiento iniciada por la Terminación Portátil: Temporizador `<CC.03>`	Elaborar un patrón de comportamiento, pero no invocarlo en el diagrama del caso de uso sino indicarlo textualmente. Nombre genérico: `ReinicioNombreTemporizador` (ej: `ReinicioCC03`).
Vencimiento de un temporizador	La activación y la parada del temporizador tienen lugar en diagramas distintos	Asignación temporal de identidad: Temporizador `<MM_ident.1>`	Definir en cada uno de los subdiagramas un temporizador de igual nombre.
Vencimiento de un temporizador	Se produce en un subdiagrama invocado por varios diagramas	Rechazo de autentificación: Temporizadores `<MM_auth.1>`, `<MM_auth.2>` y `<MM_key.1>`	Utilizar comentarios textuales para hacer la equivalencia entre los nombres de los temporizadores de cada diagrama
Vencimiento de un temporizador	Se puede detener en varios subdiagramas de ejecución opcional	Establecimiento de una llamada iniciada por la Terminación Portátil: Temporizador `<CC.03>`	Asumir que cada subdiagrama se ejecuta como si los demás no lo hicieran. Cada posible vencimiento del temporizador se debe tomar como excluyente respecto de los otros.
Reutilización de diagramas	Diagramas con la misma secuencia de acciones, actuando las mismas entidades, pero los roles de cada entidad se encuentran intercambiados	Autentificación de la Terminación Fija y de la Terminación Portátil	Utilizar, en la invocación del diagrama, un apóstrofe tras el nombre del diagrama. El apóstrofe representa el intercambio de roles entre las entidades.

La Figura 7.9 muestra la vista que ofrece la herramienta tras la inclusión de los modelos elaborados en esta actividad (sólo se muestra la sección correspondiente a esta actividad). En primer lugar aparece el documento textual con las aclaraciones a la

SPICUM servicio de publicaciones

notación MSC; a continuación, se muestran los diagramas de casos de uso y patrones de comportamiento y, finalmente, el modelo de objetos del análisis.

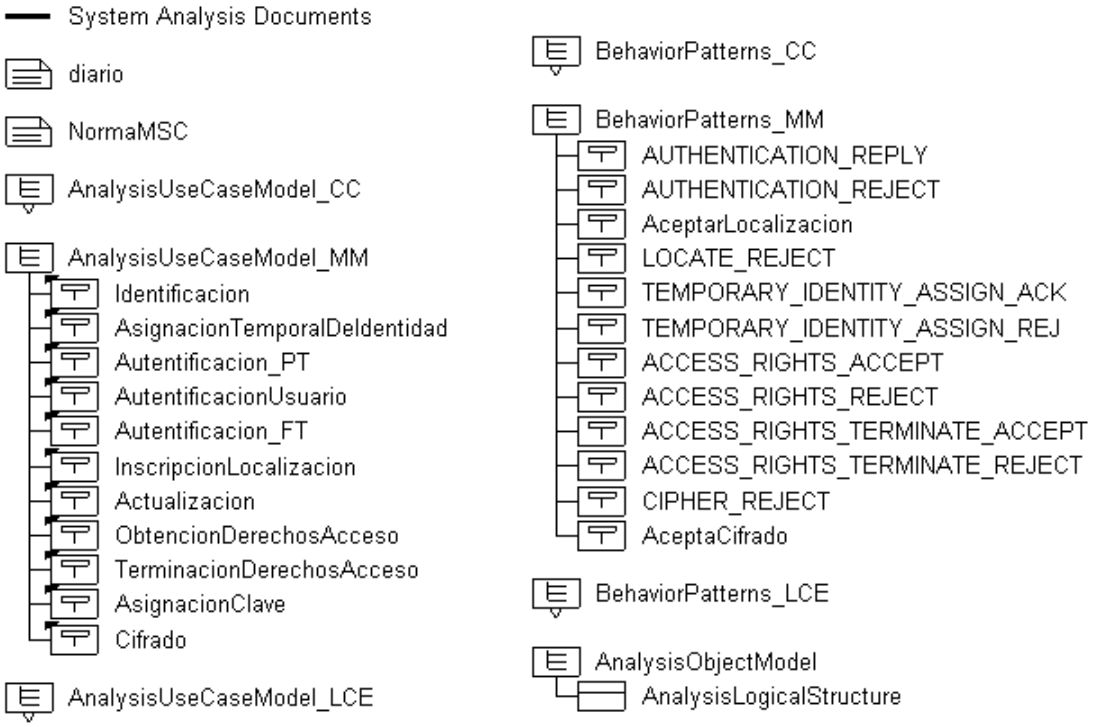


Figura 7.9: Vista del organizador de documentos tras la actividad de análisis del sistema.

7.4 Diseño del Sistema

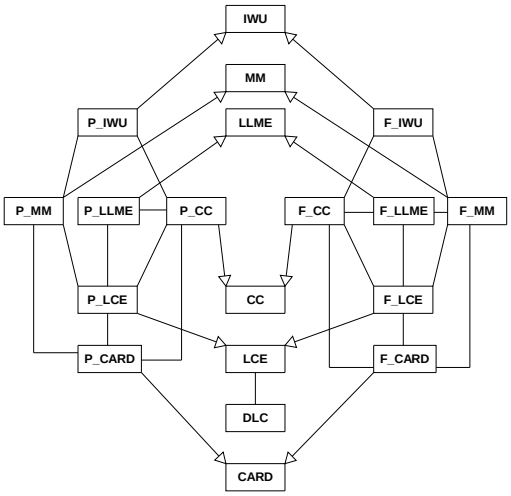
El objetivo de esta actividad es definir la arquitectura del sistema, incluyendo las interfaces entre las distintas partes que lo componen. Aunque la metodología SOMT propone comenzar esta actividad con la definición de la estructura de módulos del diseño, se ha preferido refinar el modelo de objetos del análisis, que en su estado actual es demasiado limitado para ésta y posteriores actividades, generando un nuevo modelo de objetos del diseño. Este modelo ofrece una vista funcional del sistema e incluye aspectos como los atributos de los objetos.

El nuevo modelo (Figura 7.10-a) incorpora los objetos que aparecían en el modelo resultante de la actividad de análisis del sistema, pero también incorpora nuevas entidades necesarias para la operación del sistema como, por ejemplo, una clase que almacene la información de identidades y derechos de acceso necesarios. Aquellos objetos que quedaron completamente definidos en el modelo anterior se han obviado en éste. También se han modificado algunas de las relaciones entre objetos para obtener una estructura más cercana a la realidad, por ejemplo, se ha eliminado la comunicación directa entre las entidades CC.

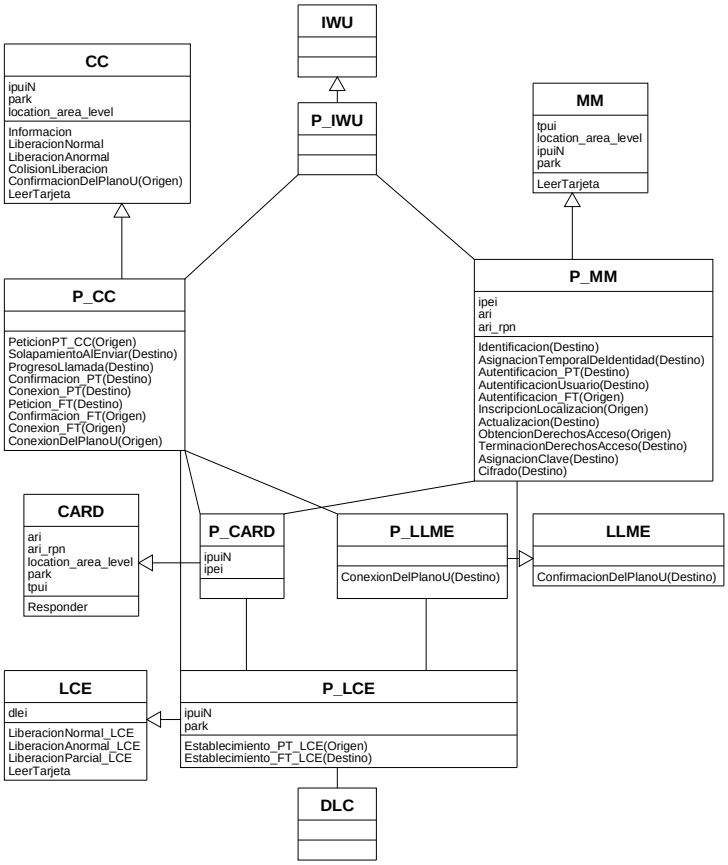
En los diagramas expandidos del modelo de objetos para la Terminación Portátil (Figura 7.10-b) y para la Terminación Fija se incluyen los atributos contenidos en cada objeto. La asignación de atributos a una clase genérica o a una clase particular de una Terminación se ha realizado siguiendo el mismo criterio que para las operaciones en la

SPICUM servicio de publicaciones

actividad anterior. Las únicas operaciones que se han añadido al nuevo modelo son las relacionadas con las nuevas clases; en este caso, la clase que modela la tarjeta de identidades (operación *LeerTarjeta* de la entidad LCE).



(a)



(b)

Figura 7.10: Diagramas (a) plegado y (b) expandido del modelo de objetos del diseño para la Terminación Portátil.

SPICUM servicio de publicaciones

Las nuevas entidades incluidas en estos modelos se han conectado con sus correspondientes conceptos presentes en el diccionario de datos. Estos conceptos se han incorporado durante esta actividad, ya que la elaboración del diccionario de datos es un proceso continuo a lo largo del diseño. El resto de los elementos de estos modelos se han conectado con los elementos correspondientes de modelos anteriores.

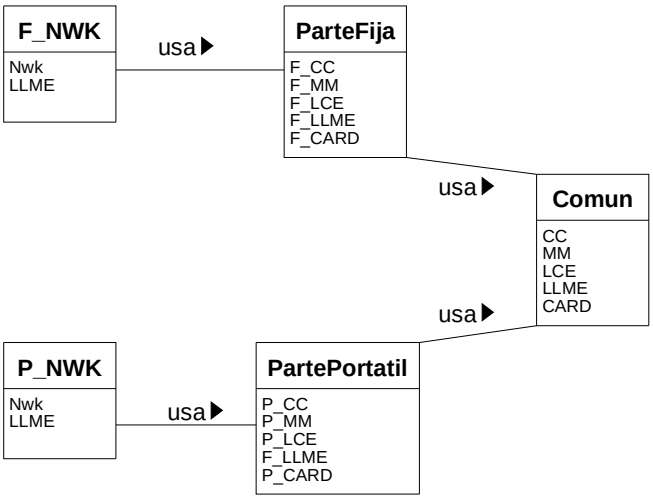


Figura 7.11: Estructura de módulos del diseño.

La estructura de módulos del diseño se genera a partir de la información aportada por el modelo de objetos del diseño, complementada con consideraciones sobre la implementación final del sistema como, por ejemplo, el lenguaje a utilizar en el modelado, los requisitos del resultado final, etc. Esta estructura está compuesta por los módulos indicados en la Figura 7.11. En el caso de diseños software, como éste, los componentes de esta estructura son módulos de código fuente; a la hora de construir el modelo se debe tener en cuenta la estrategia de implementación de cada uno de estos módulos y la reutilización de estos u otros módulos ya existentes.

Los módulos de mayor nivel son `F_NWK` y `P_NWK`; incluyen tanto la entidad del Nivel de Red como entidades que colaboran con él. Estos módulos contienen información estructural del sistema, mientras que el resto de módulos contienen información funcional. Los módulos `ParteFija` y `PartePortatil` recogen las descripciones funcionales de cada entidad, particularizadas para el rol que desempeñan. Los aspectos comunes de estas entidades se engloban en el módulo `Comun`. Los objetos del modelo de objetos del diseño se enlazan con los módulos de la estructura de módulos del diseño. Muchos de estos objetos tienen un doble enlace porque se unen tanto a los módulos de mayor nivel, `F_NWK` o `P_NWK`, como a los elementos `ParteFija` o `PartePortatil`.

La definición de la arquitectura es un paso crítico, al ser la base sobre la que se elaborará el modelo definitivo del sistema. Utiliza como punto de partida la estructura de módulos del diseño y, a través de ella, el modelo de objetos del diseño. En este paso será importante definir no sólo la estructura de bloques, sino también las interfaces entre cada uno de ellos[9]. Este modelo hace uso, por primera vez, del lenguaje elegido para el modelo final, en este caso SDL.

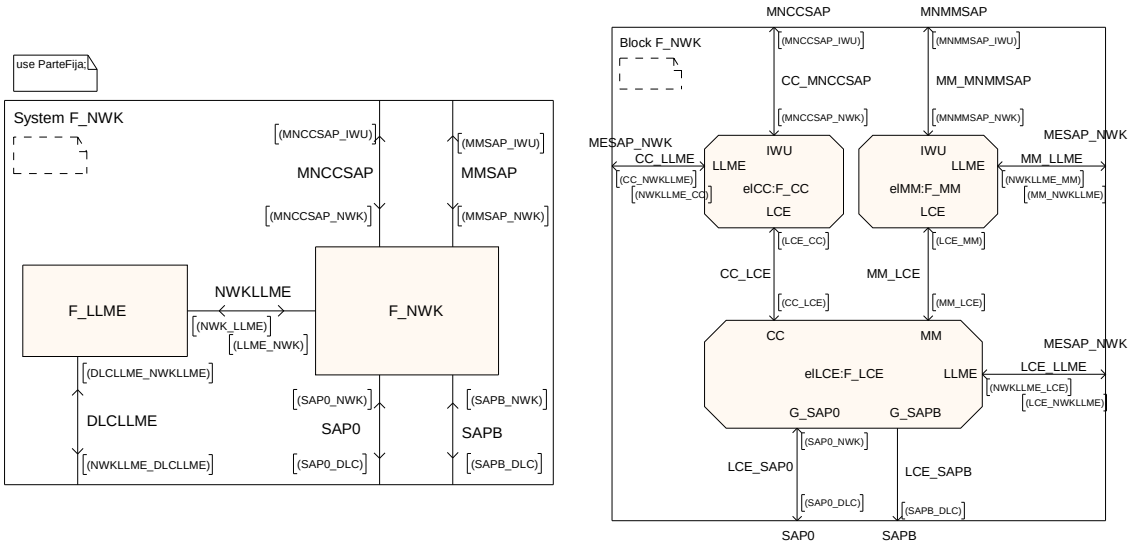
[9] Esto es especialmente importante cuando se divide el trabajo en equipos.

SPICUM servicio de publicaciones

En nuestra adaptación de la metodología SOMT, la definición de la arquitectura del sistema se ha considerado una tarea algo más amplia que la propuesta original, Se han incluido algunas tareas que pertenecían a la actividad de diseño de objetos, buscando que el resultado fuera un modelo que mostrara toda la información presente en el modelo de objetos del diseño. De esta forma, se deja para la siguiente actividad únicamente la definición de los comportamientos de cada elemento, pero no la introducción de nuevos elementos.

Los objetos activos de la estructura de módulos del diseño se han modelado como bloques (tipos de bloques) o procesos (tipos de procesos) según las relaciones de composición; sus atributos corresponden a variables mientras que sus operaciones corresponden a procedimientos remotos o puertas de señales según sean operaciones síncronas o asíncronas, respectivamente. Los objetos pasivos se han modelado como tipos de datos con atributos y operaciones.

Los módulos de nivel superior, con información estructural, se han modelado como sistemas (Figura 7.12-a), mientras que los elementos que contienen se han modelado como bloques. Aunque no aparecen en la estructura, el comportamiento de las entidades adyacentes al Nivel de Red (`IWU` y `DLC`) se considera asumido por el entorno. El bloque `F_NWK`, que representa el plano de señalización del Nivel de Red contiene tres entidades (`CC`, `MM` y `LCE`) modeladas cada una de ellas como un proceso (Figura 7.12-b). Por su parte, el bloque `LLME` contiene las entidades `LLME` y `Card` siendo cada una de ellas un único proceso.



(a) *(b)*

Figura 7.12: Diagramas SDL para la Terminación Fija de (a) el sistema y (b) la entidad del Nivel de Red.

Los módulos que no son de nivel superior, que ofrecen información funcional, se han modelado como paquetes (Figura 7.13); su contenido se obtiene a partir de la relación de componentes que aparece en cada módulo de la estructura. La clase correspondiente a cada componente se ha modelado como un tipo de proceso. El mecanismo de herencia se ha utilizado para heredar los comportamientos de las clases definidas en el paquete Comun. En esta descripción, se han utilizado procedimientos allí donde se repetían

SPICUM servicio de publicaciones

acciones en varios puntos o donde se ha preferido simplificar la descripción de algún diagrama.

Al definir las interfaces, se pueden distinguir dos aspectos: el aspecto estático, que define las operaciones y servicios ofrecidos por un bloque, y el aspecto dinámico, que define cómo colaboran los bloques para lograr un objetivo común. En la definición de la arquitectura se modela exclusivamente el aspecto estático; por ello la definición de los bloques incluirá las operaciones identificadas en tareas anteriores.

El aspecto dinámico de las interfaces se representaría en el modelo de casos de uso del diseño que propone la metodología SOMT. En el caso de desarrollos realizados por diferentes equipos, este modelo ofrece también una visión de cómo deben cooperar los bloques y de sus responsabilidades respectivas. Sin embargo, en nuestro desarrollo, se ha decidido prescindir de la elaboración de este modelo ya que el modelo de casos de uso del análisis, al haber sido elaborado a partir de los propios estándares, contiene un nivel de detalle suficiente y consistente con la definición estática de las interfaces.

Cada elemento de la definición de la arquitectura del sistema se ha enlazado a su correspondiente homólogo en la estructura de módulos del diseño, manteniendo así la información sobre el flujo del diseño.

La Figura 7.13 muestra la vista que ofrece la herramienta tras la inclusión de los modelos elaborados en esta actividad (sólo se muestra la sección correspondiente a esta actividad). En primer lugar aparece el modelo de objetos del diseño, a continuación la estructura de módulos del diseño y, finalmente, la definición de la arquitectura. Como se observa, los sistemas `F_NWK` y `P_NWK` contienen todas las entidades que estarán presentes en el sistema final. No ha sido necesario incluir documentos adicionales (textuales) a los modelos presentados.

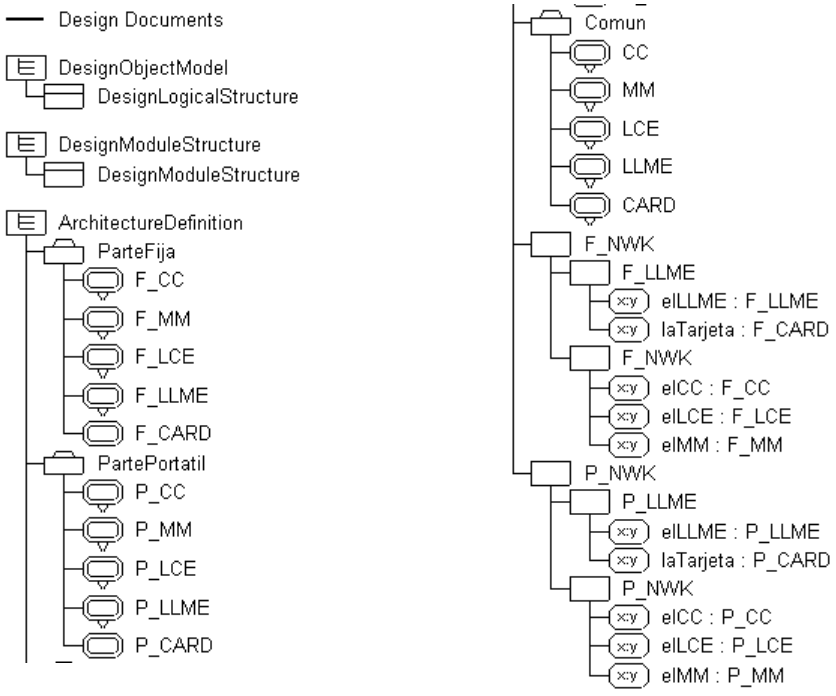


Figura 7.13: Vista del organizador de documentos tras la actividad de diseño del sistema.

SPICUM servicio de publicaciones

7.5 Diseño de Objetos

La actividad de diseño de objetos se encarga de modelar el comportamiento de los elementos presentes en la definición de la arquitectura. La metodología propone un enfoque iterativo, partiendo de un esbozo del comportamiento (primer paso) que se va refinando sucesivamente (segundo paso). La diferencia entre el primer y el segundo paso de la definición del comportamiento radica en el nivel de detalle, pero ambos tienen en consideración todos los procedimientos definidos en cada entidad.

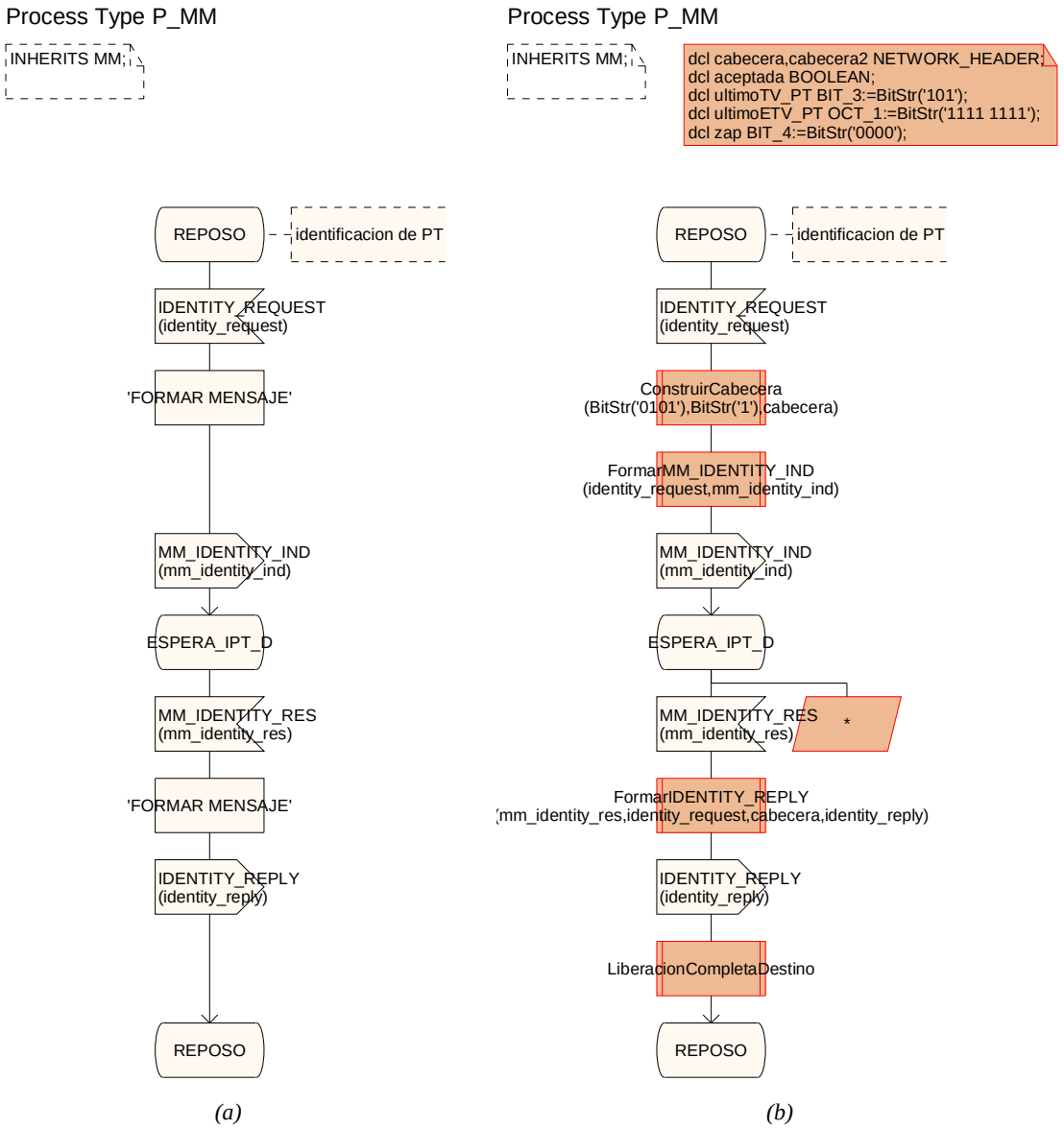


Figura 7.14: Descripción (a) intermedia y (b) definitiva del procedimiento de identificación (Terminación Portátil).

La primera tarea es la definición de las interfaces. Las señales y los tipos de datos de los parámetros que transportan aún no han sido considerados en ningún modelo anterior. Las señales correspondientes a las interfaces externas del Nivel de Red, se han generado mediante la herramienta *GenDef* (Capítulo 5, Sección 5.5.2). En el caso de las interfaces internas la definición se ha realizado manualmente; también se ha hecho así la

SPICUM servicio de publicaciones

definición de las interfaces con los bloques `LLME`, ya que no están completamente definidas en el estándar. Todas las interfaces se han definido en documentos textuales.

El primer paso en la definición del comportamiento utiliza como entrada la información presente en el modelo de casos de uso del análisis. El resultado es una descripción con un nivel de detalle intermedio, como se observa en la Figura 7.14-a para el procedimiento de identificación de la Terminación Portátil. La metodología indica que la descripción se puede estructurar por estados o por entradas; en este caso, se ha optado por la primera alternativa, ya que el conjunto de estados manejados en cualquiera de las entidades es considerablemente menor que el conjunto de las señales de entrada.

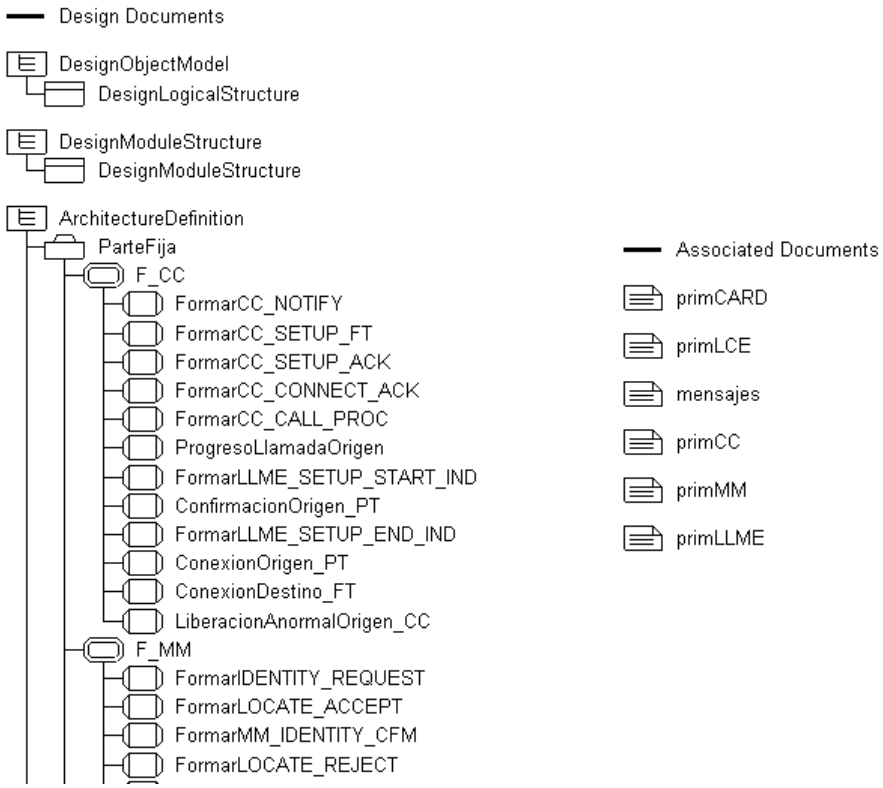


Figura 7.15: Vista del organizador de documentos tras la actividad de diseño de objetos: (a) sección Design Documents y (b) sección Associated Documents.

La información necesaria para realizar la descripción detallada[10], el segundo paso en la definición del comportamiento, se ha obtenido directamente a partir del estándar. En este paso, los elementos informales (tareas con texto informal y comentarios) son sustituidos por descripciones formales (llamadas a procedimientos y asignaciones). Dado que las entidades `CC` y `MM` hacen uso de la entidad `LCE`, cada iteración de este paso ha comenzado incrementando el detalle de esta última entidad. El resto de entidades se van detallando según su funcionalidad vaya siendo necesitada por las entidades

[10] Básicamente, el trabajo de este segundo paso consiste en la eliminación de todos los elementos informales presentes en el resultado del paso anterior. Esto incluye los valores concretos transportados en los mensajes, las comprobaciones a realizar y, en general, todas las acciones realizadas entre los intercambios de señales, las manipulaciones de temporizadores y los cambios de estado.

SPICUM servicio de publicaciones

principales. El uso de listas de estados, para modelar la respuesta a una señal en un único diagrama independientemente del estado actual de la entidad, y de procedimientos, encapsulando acciones utilizadas en varios puntos, permite simplificar los diagramas. El resultado de esta tarea, para el ejemplo indicado en el paso anterior, se muestra en la Figura 7.14-b.

Los elementos del modelo resultante de esta actividad se han conectado al diccionario de datos y a los epígrafes correspondientes del estándar; estas últimas conexiones se han realizado mediante comentarios textuales, ya que el estándar en sí no forma parte del conjunto de modelos incluidos en la herramienta. Como el diseño de objetos se ha realizado sobre el modelo de la definición de la arquitectura, los enlaces de este modelo siguen estando vigentes.

La vista que ofrece la herramienta tras la inclusión de los modelos elaborados en esta actividad es similar a la anterior, pero conteniendo todos los procedimientos empleados finalmente para modelar el comportamiento. La Figura 7.15-a muestra parcialmente esta situación. Se ha creado una sección unificada para esta actividad y la actividad de diseño del sistema, ya que se ha trabajado sobre el modelo resultante de la actividad anterior. Además, se ha creado una nueva sección, Associated Documents (Figura 7.15-b), que no pertenece a ninguna actividad concreta, sino que contiene documentos relacionados con la definición de los parámetros de las señales.

7.6 Implementación

La actividad de implementación consta en la metodología SOMT de las siguientes tareas: particionado en uno o más módulos software o hardware, adaptación de la comunicación entre el sistema SDL y su entorno, integración con el hardware y la plataforma de ejecución, generación de código y pruebas. La descripción de estas tareas se incluye en el Capítulo 8, Sección 8.5.3.3, ya que el Nivel de Red implementado se ha utilizado como parte del Emulador del Sistema Bajo Prueba en el rol de la Terminación Portátil y no como una entidad aislada.

7.7 Metodología SOMT Modificada (M-SOMT)

Como se ha ido indicando a lo largo de la descripción de las tareas realizadas en cada actividad, la metodología SOMT ha sido modificada en algunos aspectos. A la metodología resultante de la incorporación de estos cambios en la metodología original se la ha denominado metodología SOMT Modificada (M-SOMT – *Modified SOMT*) [ALBA00a]. Esta metodología mantiene las ventajas de la metodología SOMT, pero está adaptada al diseño en SDL de sistemas basados en estándares. El conjunto de modelos y actividades de la metodología M-SOMT se muestra en la Figura 7.16.

La actividad de análisis de requisitos deja de tener sentido, ya que el estudio del dominio del problema y las restricciones que impone sobre el sistema ha sido realizado durante el desarrollo del estándar. El estándar es, pues, incorporado como un documento sobre el que se basarán los modelos posteriores. Sin embargo, sí se elabora el diccionario de datos, ya que permite presentar de otra forma los conceptos presentes en el estándar y servirá como anclaje donde conectar los elementos de los modelos posteriores, permitiendo trazar el concepto hasta su origen.

La actividad de análisis del sistema se ha mantenido sin modificaciones. En ella se elaboran el modelo de casos de uso del análisis, que presenta una visión global de cómo

SPICUM servicio de publicaciones

se comportan las entidades del sistema y el modelo de objetos del análisis, que ofrece un primer esbozo de cuál será la estructura del sistema.

En la actividad de diseño del sistema sí se han introducido numerosos cambios. No sólo se han eliminado modelos que se han considerado no relevantes, sino que se ha incorporado el modelo de objetos del diseño. Tanto este modelo como la estructura de módulos del diseño se expresan mediante la notación del modelo de objetos del análisis. Por su parte, el modelo de definición de la arquitectura utiliza ya el lenguaje que se empleará en el modelo abstracto final, SDL. Esta definición constituye un paso intermedio en la elaboración del modelo definitivo.

Se ha decidido que, a diferencia de la metodología SOMT, la definición de la arquitectura incluya la definición de todos los elementos derivados del modelo de objetos del diseño, lo que engloba no sólo a los bloques, como originalmente, sino también a los procesos y tipos de procesos especializados de las clases presentes en el modelo indicado. La metodología propone diversas estructuras posibles para cada tipo de objeto, según su función, pero no es posible automatizar el paso entre ambos modelos, al estar sujeto a numerosas decisiones de diseño.

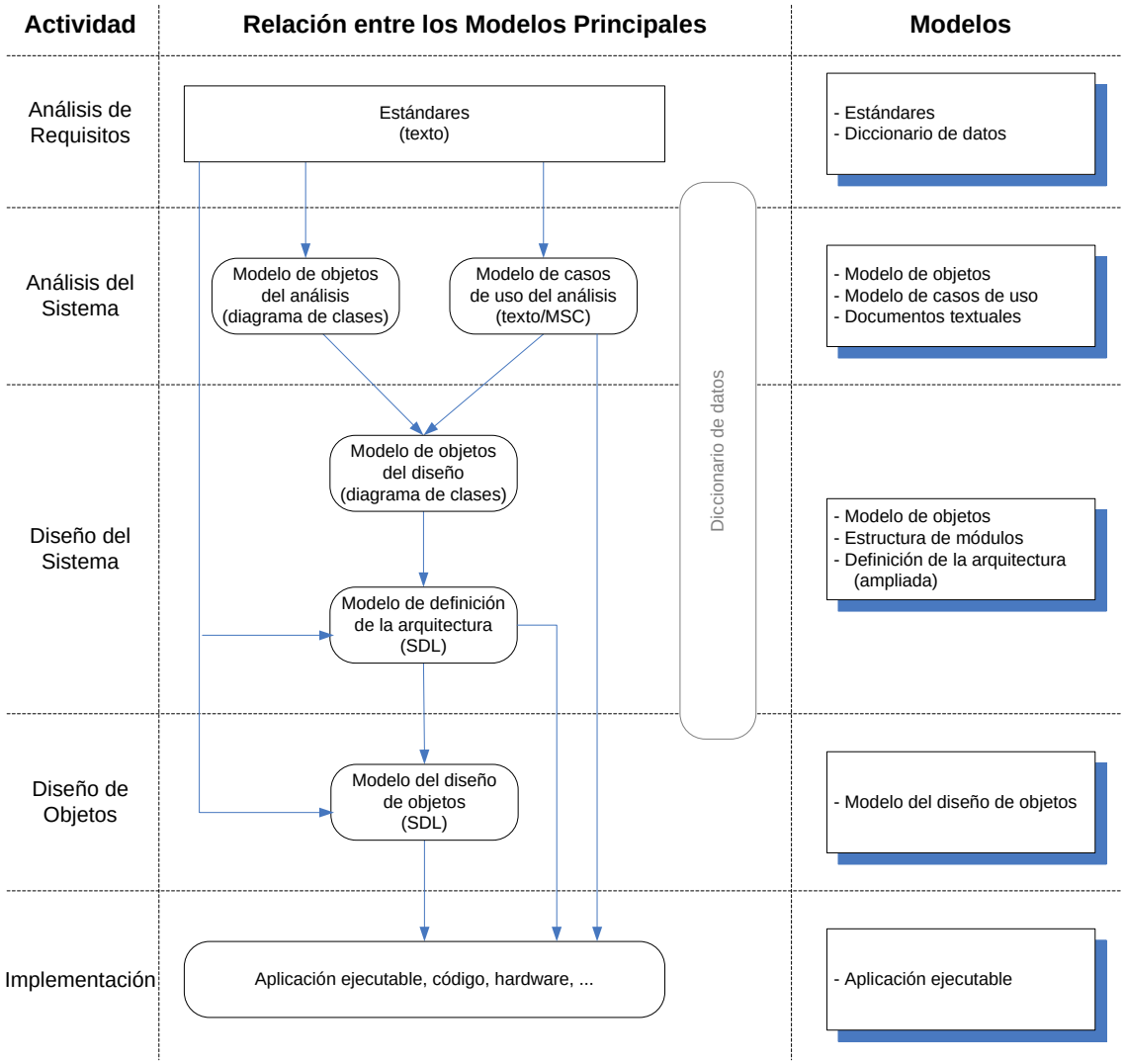


Figura 7.16: Actividades y modelos de la metodología M-SOMT.

SPICUM servicio de publicaciones

La actividad de diseño de objetos consiste en la descripción en SDL del comportamiento de cada elemento de la arquitectura. Durante este modelado, surgirán nuevos procedimientos que serán incorporados a la arquitectura. Como resultado se obtiene un modelo en SDL que ofrece la funcionalidad exigida en el análisis de requisitos y cumple las restricciones impuestas.

La última y quinta actividad, la implementación, depende en gran medida de las características de la plataforma final. La metodología original sólo ofrece algunas indicaciones muy generales. Por ello, se ha considerado adecuada su descripción. En la implementación que utiliza el Nivel de Red modelado, en general, se han realizado las tareas propuestas en el orden sugerido, como se puede observar en el Capítulo 8.

La metodología resultante presenta las ventajas asociadas al uso del enfoque orientado a objetos, así como las derivadas de la utilización de lenguajes formales. La orientación a objetos permite la reutilización del código de una forma más clara e intuitiva que en un desarrollo tradicional y facilita una mayor estructuración del modelo. En conjunto, esto redunda en una reducción del tiempo dedicado al mismo, a pesar de la mayor inversión inicial necesaria; la mayor ganancia de tiempo se obtiene en el diseño detallado del comportamiento de los objetos. Se ha estimado que esta reducción puede alcanzar un 20% para un sistema como el considerado en este análisis [ALBA00b].

7.8 Conclusiones

Una de las ventajas de la metodología SOMT estriba en su integración del enfoque orientado a objetos con el uso de lenguajes formales, en concreto, SDL. La división de actividades y los modelos que propone no suponen una ventaja por sí mismos, pero sí lo es la utilización de la información representada en cada uno de ellos para la elaboración de los modelos posteriores y la trazabilidad que se aporta al proceso. Cada uno de los modelos ofrece un aspecto complementario del sistema, con el objetivo global de la descripción formal realizada en la actividad de diseño de objetos.

Cuando se trabaja en un desarrollo basado en un estándar, las tareas de la actividad de análisis de requisitos carecen en su mayoría de utilidad, ya que han sido realizadas previamente a la definición de dicho estándar y es suficiente con incorporar los documentos correspondientes. Aún así, se ha considerado adecuado mantener la realización del diccionario de datos pues su utilidad se prolonga durante todo el proceso de diseño.

Los modelos de la actividad de diseño del sistema son los que más modificaciones sufren. Se ha incorporado un nuevo modelo, el modelo de objetos del diseño, y se ha ampliado el ámbito de la definición de la arquitectura, exigiendo la definición de todos los elementos del sistema, no sólo de sus bloques. Sin embargo, se ha prescindido del modelo de casos de uso de esta actividad porque los modelos anteriores, al estar obtenidos a partir de la norma, son ya consistentes y suficientemente detallados.

Integrando estas modificaciones en la metodología original se ha definido una metodología, M-SOMT, que, manteniendo las ventajas de la primigenia, adapta las tareas a realizar a las características del diseño de sistemas basados en un estándar. El resultado es un proceso con una estructura donde se ha reforzado el uso de modelos previos para la elaboración de los posteriores; no obstante, se mantiene el enfoque iterativo en el diseño. Esta metodología ha sido aplicada al diseño del Nivel de Red del sistema DECT que se ha utilizado en la comprobación de los Sistemas de Pruebas descritos en el Capítulo 8.

SPICUM servicio de publicaciones

SECCIÓN II

IMPLEMENTACIONES

SPICUM
servicio de publicaciones

SPICUM
servicio de publicaciones

En esta Sección se describe la implementación de Sistemas de Pruebas para las tecnologías DECT, Bluetooth y UMTS siguiendo la Metodología de Diseño presentada. A lo largo de los diseños que aquí se presentan se puede observar la evolución seguida. Mientras que los Sistemas de Pruebas para DECT se han realizado en un entorno académico con el objetivo de validar la Metodología de Diseño, la arquitectura genérica y las herramientas de soporte, los Sistemas diseñados para Bluetooth y UMTS han aplicado estas propuestas a un entorno industrial, lo que ha provocado modificaciones en las propuestas originales.

Los cambios que se han ido produciendo en la filosofía subyacente de aplicación de la Metodología de Pruebas de Conformidad OSI han creado la necesidad de evolucionar los distintos elementos del diseño. Esto se ha producido, sobre todo, en el área de los codificadores, donde se ha pasado de utilizar tipos nativos TTCN y sintaxis de transferencia ASCII al uso de tipos de datos ASN.1 y sintaxis de transferencia PER, con los cambios que ello requiere en las herramientas de soporte desarrolladas.

Por otro lado, el interés en las pruebas de equipos se ha ido trasladando de las pruebas de conformidad a las pruebas de interoperatividad. Esto ha quedado reflejado en este trabajo con la implementación de Sistemas de Pruebas de interoperatividad para Bluetooth y, con una mayor formalización del proceso de diseño, UMTS.

En primer lugar, se describe la implementación de Sistemas de Pruebas de conformidad para equipos DECT. La descripción se presenta con un alto grado de detalle con objeto de mostrar la complejidad del proceso. Estos diseños han servido para validar la Metodología y las herramientas asociadas a la misma. Han permitido, también, evaluar algunas de las alternativas indicadas en la Metodología para realizar la comunicación externa del Módulo de Protocolos como son la ubicación de los codificadores y la comunicación con el Módulo de Capa Física. Los sistemas resultantes han sido ejecutados sobre las plataformas Unix, Linux, Windows y DSP/BIOS.

Los resultados obtenidos han sido aplicados al diseño de un Sistema de Pruebas comercial de protocolos para equipos Bluetooth dentro de la colaboración con la industria. Ello ha permitido mejorar las herramientas y componentes utilizados con las sugerencias del grupo de diseño de la empresa. El Sistema de Pruebas BITE ha sido el más vendido en todo el mundo. Se ha demostrado cómo la arquitectura propuesta no está ligada al uso de herramientas comerciales concretas. Además, se ha comprobado que es también aplicable a Sistemas de Pruebas de interoperatividad.

La colaboración con la industria ha continuado con la construcción de Sistemas de Pruebas de conformidad de UMTS (familia MINT). El resultado ha sido también utilizado como Unidad de Señalización en Sistemas de Pruebas radio. La evolución en el modelado de las pruebas de conformidad y en las Especificaciones de Sistema han llevado a integrar la notación ASN.1 y la sintaxis de transferencia PER en las herramientas propias. El diseño del Módulo de Protocolos se ha mejorado, empleando el mismo componente para las pruebas de cualquier nivel; la configuración necesaria se selecciona dinámicamente. Además, se ha avanzado en la formalización del diseño de Sistemas de Pruebas de interoperatividad, integrando herramientas comerciales y adaptando las propias.

Por último, se describe el diseño de Sistemas de Pruebas radio para las tecnologías Bluetooth y UMTS. Con ello, se comprueba cómo el uso de una misma notación para las pruebas de protocolos y las pruebas radio permite reutilizar los componentes de la arquitectura y reducir los costes. También se demuestra el uso práctico de la interfaz de primitivas propuesta para el control de la instrumentación.

SPICUM servicio de publicaciones

SPICUM
servicio de publicaciones

Capítulo

8

"Eran paisajes hermosos, de localizaciones en la superficie o muy hábitats muy grandes. Cada uno de ellos era irreal en iluminación y geometría, pero preciso hasta el detalle de las briznas de hierba."

Pham Nuwen, Un Abismo en el Cielo

CAPÍTULO 8: SISTEMAS DE PRUEBAS DECT

La Metodología presentada en los capítulos anteriores se ha aplicado a la construcción de un Sistema de Pruebas para equipos del sistema de comunicaciones DECT (*Digital Enhanced Cordless Telecommunications*). DECT es un sistema de telefonía digital inalámbrica[1] de corto alcance desplegada a nivel mundial [ETR 183] incluida en la familia IMT-2000 del ITU, donde recibe la denominación IMT-FT (*International Mobile Telecommunications – Frequency Time*) [PEJA02]. Está diseñado para operar en entornos públicos, residenciales o empresariales. La comunicación (Figura 8.1) se realiza desde una Terminación Portátil (equipo móvil) (PT – *Portable Termination*) a través de una Terminación Fija (estación base) (FT – *Fixed Termination*) que actúa de pasarela con las redes públicas de telefonía y datos (GSM, X.25, ISDN, UMTS). Las características más relevantes en las que se asienta esta tecnología son la selección dinámica del canal, que hace innecesaria la coordinación entre equipos instalados, el uso de espectro sin licencia[2], la posibilidad de desplazarse entre distintas zonas de cobertura, y la seguridad que proporciona a las comunicaciones.

La decisión de elegir el sistema DECT y no otro se basó en varios factores. En primer lugar, en el momento de tomar la decisión (año 1997) se trataba de un sistema ya maduro, con una amplia cuota de mercado en su segmento en numerosos países [VOLL94], incluso en competencia con los sistemas PACS (*Personal Access Communications System*) y PHS (*Personal Handy-phone System*) ([COX95], [NOER96], [TUTT92],). En segundo lugar, DECT ha sido, junto con GSM, de los primeros sistemas en aplicar la metodología de pruebas de conformidad extensivamente. En el año 1996 se termina la primera versión de las Pruebas de Conformidad del sistema. Estas pruebas incluían los Niveles de Red (NWK – *NetWorK*), Enlace (DLC – *Data Link Control*) y Acceso al Medio (MAC – *Medium Access Control*) tanto para la Terminación Portátil

[1] Como tecnología, pertenece a la segunda generación de sistemas de telefonía digital sin hilos [PADG95].

[2] Es el único miembro de la familia IMT-2000 con esta característica.

SPICUM servicio de publicaciones

como para la Terminación Fija del sistema[3]. En tercer lugar, existían diversas opciones comerciales que implementaban las capas inferiores de la interfaz aire; de otra forma, hubiera sido necesario diseñar el hardware requerido para dichas capas incluidas en el Sistema de Pruebas.

DECT se ha convertido en la tecnología dominante en el mercado inalámbrico residencial y en el segmento de las PABX (*Private Automatic Branch Exchange*) [DECT] Muestra de la importancia de este sistema para ETSI es el hecho de que exista un proyecto independiente dentro del organismo estandarizador para mantener y desarrollar los estándares de dicho sistema. Su flexibilidad y facilidad de planificación[4] han facilitado su uso en múltiples aplicaciones como redes de área local sin hilos ([MORE95], [BERW96]), acceso radio en el bucle de abonado ([REIS95], [WLL600]) o centralitas sin hilos para oficinas ([NATI01], [PRIX02])[5]. La tecnología continúa evolucionando con la nueva generación de equipos DECT (*New Generation DECT™*) [TR 102 570].

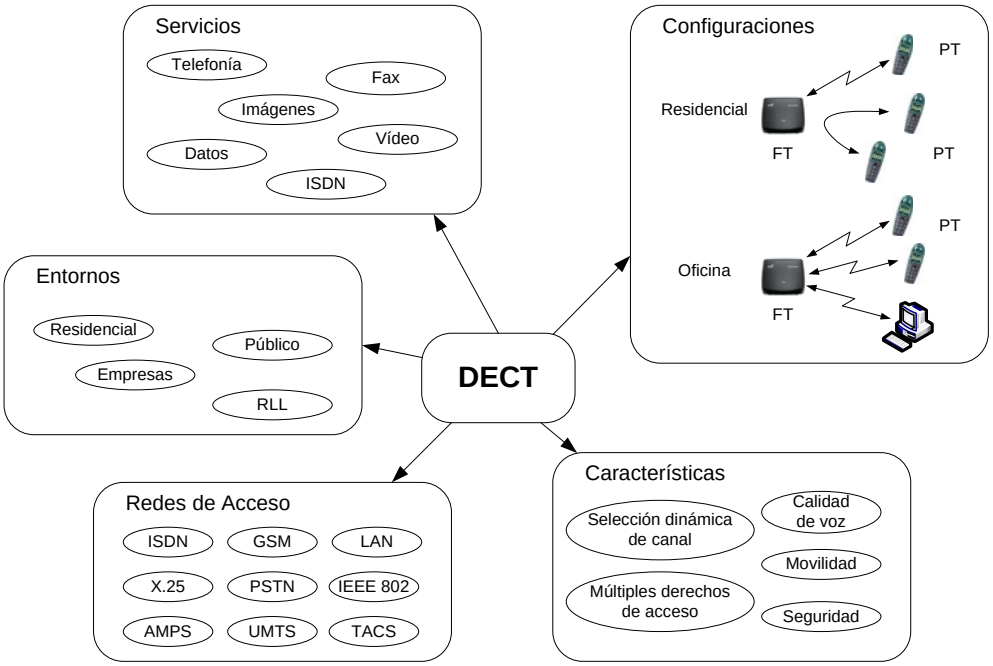


Figura 8.1: Resumen de las aplicaciones y características del sistema de comunicaciones DECT. [ETR 178]

Este trabajo ha sido realizado dentro de un proyecto financiado por la Unión Europea [PROY99]. A lo largo del mismo, se han desarrollado, de acuerdo con la Metodología de Diseño expuesta, Sistemas de Pruebas para los Niveles de Red y de Enlace del sistema DECT. La implementación de estos Sistemas permite validar la Metodología de Diseño presentada en el Capítulo 4. Las Pruebas que se han implementado son las estándar para el perfil GAP [ETS 300 176-2], que proporciona la funcionalidad básica para aplicaciones de telefonía de voz. En el Apéndice F se describe el Sistema de Comunicaciones DECT; el conjunto de las normas que definen este sistema se lista en el Apéndice G.

[3] Las pruebas de la capa DLC son las mismas para la Terminación Portátil y la Terminación Fija.

[4] Por el uso de técnicas de asignación dinámica de canales.

[5] Se puede obtener información sobre productos comerciales que utilizan la tecnología DECT en [DEWE07].

SPICUM servicio de publicaciones

A lo largo de este capítulo se detalla el proceso seguido para la construcción de los mencionados Sistemas de Pruebas, es decir, los pasos para su diseño, modelado, simulación y validación. Para facilitar la legibilidad, se habla de forma habitual haciendo referencia a un único Sistema de Pruebas; debe entenderse que la descripción corresponde a todos los Sistemas de Pruebas construidos. La descripción que se realiza posee un alto grado de detalle con objeto de mostrar la complejidad del proceso. Se ha organizado siguiendo las fases de la Metodología presentada en el Capítulo 4 (ver Figura 4.3); para una mejor comprensión, algunas de la figuras presentes en dicho capítulo se repiten en éste. Igualmente, junto al título de cada fase se incluye una figura reducida que indica su ubicación en la secuencia de actividades.

En primer lugar, se fijan los objetivos que deben cumplir los Sistemas de Pruebas. Seguidamente, se explica cómo se ha realizado el diseño, describiendo cada fase en una sección diferente (de la 8.2 a la 8.6); a lo largo de esta exposición se comentará cómo se han resuelto aquellas particularidades que han surgido a lo largo del camino. Finalmente, se ha incluido una breve descripción de los Sistemas de Pruebas comerciales para DECT.

8.1 Objetivos de los Sistema de Pruebas

Los Sistemas de Pruebas que se presentan en este capítulo permiten la ejecución de los Casos de Prueba del Perfil de Acceso Genérico (GAP) [ETS 300 444] publicados por ETSI para los Niveles DLC de las Terminaciones Fija y Portátil y el Nivel NWK de la Terminación Portátil del sistema DECT.

Las Especificaciones de Base del Sistema DECT se denominan Interfaz Común (CI – *Common Interface*) y ocupan las partes 1 a 8 del estándar [ETS 300 175]. La Interfaz Común define la arquitectura y la funcionalidad necesarias de un equipo DECT para que los servicios de voz y datos de equipos distintos puedan operar conjuntamente. En el Apéndice F se describe la Interfaz Común con objeto de facilitar la comprensión de este Capítulo.

El Perfil de Acceso Genérico (GAP – *Generic Access Profile*) [ETS 300 444] incluye "*los requisitos mínimos para establecer, mantener y liberar conexiones de voz a 3,1 kHz entre una Terminación Fija y una Terminación Portátil con los derechos de acceso adecuados independientemente de si la Parte Fija proporciona servicios residenciales, de empresa o de acceso público*", estando enfocado hacia áreas como la telefonía doméstica o las centralitas de empresa. Este perfil es la base de otros perfiles que hacen uso de servicios de voz. Una breve descripción del perfil GAP se incluye al final de Apéndice F.

La funcionalidad que verifican los Sistemas de Pruebas es la indicada en [ETS 300 494-1], con la salvedad de las limitaciones derivadas de las tarjetas empleadas para implementar el Módulo de Capa Física, que se pueden resumir en la carencia de la funcionalidad de cifrado y traspaso entre estaciones base. La funcionalidad implementada es la siguiente:

- Nivel DLC: Servicios del plano de control clase A y del plano de usuario clase 0.
    - Servicio de difusión.
    - Establecimiento, mantenimiento y liberación de enlaces.

SPICUM servicio de publicaciones

- Transferencia de usuario extremo a extremo, con servicio confirmado y no confirmado, de una y múltiples tramas.

- Nivel NWK: Servicios básicos de control de la llamada, gestión de la movilidad y control del enlace para comunicación básica de voz.
  - Establecimiento, mantenimiento y liberación de llamadas en modo circuito conmutado.
  - Gestión de identidades, autentificación, localización y registro en una celda.
  - Negociación y modificación de servicios, gestión de recursos y coordinación de los procedimientos de movilidad.

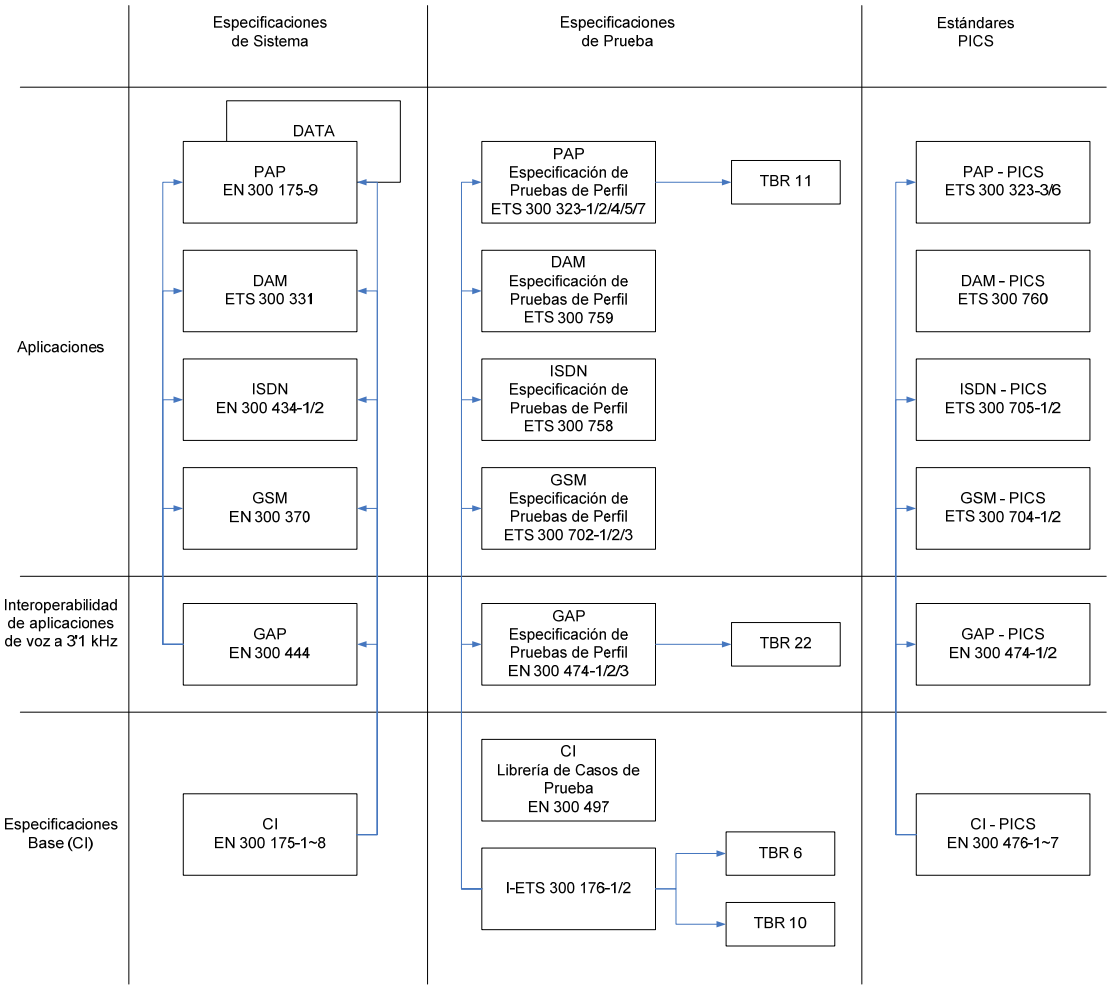


Figura 8.2: Esquema resumen de los estándares DECT [ETR 183].

8.2 Documentación

El sistema DECT engloba casi dos centenares de documentos entre estándares (EN – *European Standard*, ES – *ETSI Standard*), especificaciones técnicas (TS – *ETSI Technical Specification*) e informes técnicos (TR – *Technical Report*). La Figura 8.2 muestra un resumen de los documentos más relevantes del Sistema DECT y las

SPICUM servicio de publicaciones

relaciones entre ellos desde un punto de vista genérico. A continuación se comentan algunos, entrando más en profundidad cuando sea necesario a lo largo de este trabajo.

Tabla 8.1: Lista de Especificaciones de Sistema para DECT.

Documento	Título
Visión Global	
ETR 178	Digital Enhanced Cordless Telecommunications (DECT); A High Level Guide to the DECT Standardization
ETR 043	Digital Enhanced Cordless Telecommunications (DECT); Common Interface (CI); Services and facilities requirements specification
ETR 056	Digital Enhanced Cordless Telecommunications (DECT); System description document
Especificaciones Base	
TBR 006	Digital Enhanced Cordless Telecommunications (DECT); General terminal attachment requirements
TBR 010	Digital Enhanced Cordless Telecommunications (DECT); General Terminal Attachment Requirements; Telephony Applications
ETS 300 175-1	Digital Enhanced Cordless Telecommunications (DECT); Common Interface (CI); Part 1: Overview
ETS 300 175-2	Digital Enhanced Cordless Telecommunications (DECT); Common Interface (CI); Part 2: Physical Layer (PHL)
ETS 300 175-3	Digital Enhanced Cordless Telecommunications (DECT); Common Interface (CI); Part 3: Medium Access Control (MAC) layer
ETS 300 175-4	Digital Enhanced Cordless Telecommunications (DECT); Common Interface (CI); Part 4: Data Link Control (DLC) layer
ETS 300 175-5	Digital Enhanced Cordless Telecommunications (DECT); Common Interface (CI); Part 5: Network (NWK) layer
ETS 300 175-6	Digital Enhanced Cordless Telecommunications (DECT); Common Interface (CI); Part 6: Identities and addressing
ETS 300 175-7	Digital Enhanced Cordless Telecommunications (DECT); Common Interface (CI); Part 7: Security features
ETS 300 175-8	Digital Enhanced Cordless Telecommunications (DECT); Common Interface (CI); Part 8: Speech coding and transmission
Perfiles	
ETS 300 444	Digital Enhanced Cordless Telecommunications (DECT); Generic Access Profile (GAP)

8.2.1 Especificaciones de Sistema

ETSI ofrece una visión general del Sistema DECT y sus componentes en los documentos técnicos [ETR 056] y [ETR 178]. El primer documento, [ETR 056] (*System Description Document*), describe el Sistema DECT desde un punto de vista de sistema, incluyendo la arquitectura de protocolos, las identidades de los equipos DECT así como las funciones de movilidad especificadas; también contiene una descripción del modelo y los perfiles de interconexión a otras redes, junto con una indicación de posibles aplicaciones. El segundo documento, [ETR 178] (*A High Level Guide to the DECT Standardization*), ofrece una descripción de alto nivel de los distintos componentes englobados en la tecnología DECT.

SPICUM servicio de publicaciones

Las Especificaciones Base[6] (Tabla 8.1), que describen el Interfaz Común (CI – *Common Interface*), corresponden a las partes 1 a 8 de la norma [ETS 300 175]. El perfil relacionado con los Sistemas de Pruebas a diseñar es el Perfil de Acceso Genérico (GAP) [ETS 300 444]. Las normas correspondientes aparecen también en la Tabla 8.1. Adicionalmente a estas especificaciones, el informe técnico [TBR 006] especifica los requisitos para la transmisión radio y el informe técnico [TBR 010] lo hace para las aplicaciones de telefonía.

La primera parte de la norma [ETS 300 175] ofrece una visión general del sistema y su arquitectura de protocolos. Las partes 2 a 5 especifican, respectivamente, los niveles PHL (*PHysical Layer*), MAC (*Medium Access Control*), DLC (*Data Link Control*) y NWK (*NetWorK*) (ver Apéndice F, Secciones F.2.1 a F.2.4). La parte 6 especifica las identidades y direcciones empleadas, divididas en Terminaciones Fijas, Terminaciones Portátiles, relacionadas con la conexión y relacionadas con el equipo (ver Apéndice F, Sección F.3). La séptima parte indica la seguridad a proporcionar en el sistema, incluyendo la arquitectura, los algoritmos criptográficos y su integración en los diferentes niveles. Finalmente, la parte octava se centra en los requisitos de codificación y transmisión de voz para proporcionar una conversación dúplex con un ancho de banda de 3,1 kHz.

8.2.2 Estructura y Propósito de las Pruebas

El informe técnico [ETR 183] da una visión general de la aplicación de la metodología de pruebas de conformidad al sistema DECT. En él se indican los objetivos buscados en estas pruebas, así como la organización de los documentos en que vienen descritas. El conjunto de estos documentos se muestra en la Tabla 8.2.

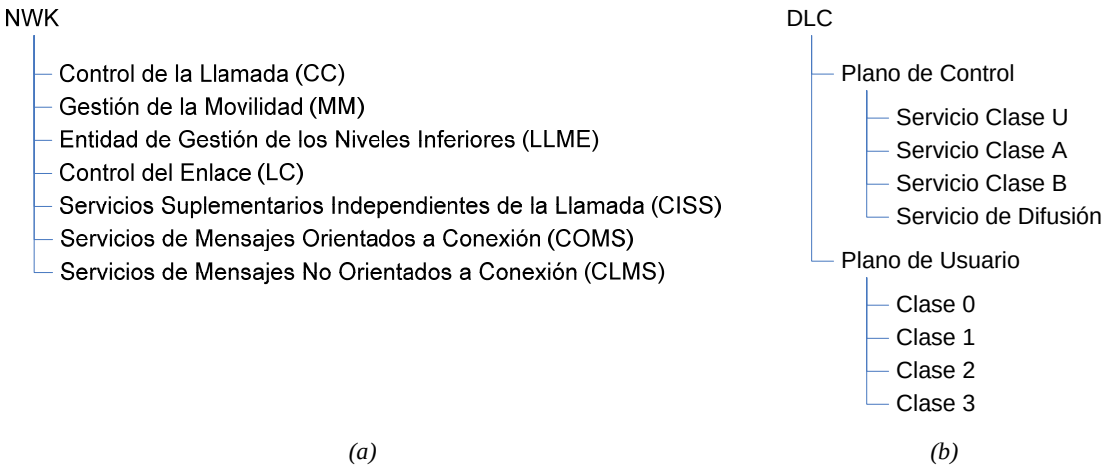


Figura 8.3: Grupos de Pruebas para los Niveles de (a) Red y (b) Control del Enlace.

Las Estructuras y Propósitos de las Pruebas de Protocolo de DECT vienen definidas como partes separadas para cada nivel; su conjunto recibe el nombre de Biblioteca de Casos de Prueba (TCL – *Test Case Library*) [EN 300 497]. Las partes 1, 4, 6 y 8 describen

[6] Aunque ya se han mencionado en la Sección anterior, se vuelven a indicar aquí las normas correspondientes a la Interfaz Común y el perfil GAP para que esta Sección sea autocontenida.

SPICUM servicio de publicaciones

la Estructura y Propósito de las Pruebas para los niveles MAC, DLC, NWK-PT[7] y NWK-FT[7], respectivamente.

Los Casos de Prueba para el Nivel de Red ([EN 300 497-6], [EN 300 497-8]) están divididos en siete grupos (Figura 8.3-a)[8]: 1) Control de la Llamada (CC – *Call Control*), 2) Gestión de la Movilidad (MM – *Mobility Management*), 3) Entidad de Gestión de los Niveles Inferiores (LLME – *Lower Layer Management Entity*), 4) Control del Enlace (LC – *Link Control*), 5) Servicios Suplementarios Independientes de la Llamada (CISS – *Call Independent Supplementary Services*), 6) Servicios de Mensajes Orientados a Conexión (COMS – *Connection Oriented Message Service*) y 7) Servicios de Mensajes No Orientados a Conexión (CLMS – *ConnectionLess Message Service*).

Tabla 8.2: Lista de documentos que describen la Estructura y Propósito de las Pruebas de conformidad para DECT.

Documento	Título
Visión Global	
ETR 183	Digital Enhanced Cordless Telecommunications (DECT); Conformance testing on DECT equipment
Estructura y Propósito de las Pruebas de Protocolo	
I-ETS 300 176	Digital Enhanced Cordless Telecommunications (DECT); Approval test specification
ETS 300 176-1	Digital Enhanced Cordless Telecommunications (DECT); Test specification; Part 1: Radio
ETS 300 176-2	Digital Enhanced Cordless Telecommunications (DECT); Test specification; Part 2: Speech
EN 300 497-1	Digital Enhanced Cordless Telecommunications (DECT); Common Interface (CI); Test Case Library (TCL); Part 1: Test Suite Structure (TSS) and Test Purposes (TP) for Medium Access Control (MAC) layer
EN 300 497-4	Digital Enhanced Cordless Telecommunications (DECT); Common Interface (CI); Test Case Library (TCL); Part 4: Test Suite Structure (TSS) and Test Purposes (TP) - Data Link Control (DLC) layer
EN 300 497-6	Digital Enhanced Cordless Telecommunications (DECT); Common Interface (CI); Test Case Library (TCL); Part 6: Test Suite Structure (TSS) and Test Purposes (TP) - Network (NWK) layer - Portable radio Termination (PT)
EN 300 497-8	Digital Enhanced Cordless Telecommunications (DECT); Common Interface (CI); Test Case Library (TCL); Part 8: Test Suite Structure (TSS) and Test Purposes (TP) - Network (NWK) layer - Fixed radio Termination (FT)
Estructura y Propósito de las Pruebas de Perfiles	
TBR 022	Radio Equipment and Systems (RES); Attachment requirements for terminal equipment for Digital Enhanced Cordless Telecommunications (DECT) Generic Access Profile (GAP) applications
TBR 022/A1	Radio Equipment and Systems (RES); Attachment requirements for terminal equipment for Digital Enhanced Cordless Telecommunications (DECT) Generic Access Profile (GAP) applications
ETS 300 494-1	Digital Enhanced Cordless Telecommunications (DECT); Generic Access Profile (GAP); Profile Test Specification (PTS); Part 1: Summary

Los Casos de Prueba para el Nivel de Control del Enlace están divididos en Pruebas para el Plano de Control y pruebas para el Plano de Usuario; ambos grupos se

[7] Para simplificar el texto, al hablar de los niveles emplearemos los sufijos '-PT' o '-FT' según queramos referirnos a la Terminación Portátil o a la Terminación Fija. En caso de no indicarse nada, entenderemos que nos estamos refiriendo a ambas terminaciones.

[8] Estos grupos son los mismos para la Terminación Portátil y la Terminación Fija.

SPICUM servicio de publicaciones

subdividen en cuatro categorías (Figura 8.3-b). Las pruebas para el Plano de Control se subdividen según el tipo de servicio en: 1) Servicio Clase U, 2) Servicio Clase A, 3) Servicio Clase B y 4) Servicio de Difusión. A su vez, las pruebas para el Plano de Usuario se subdividen según la clase de procedimiento de transmisión en: 1) Clase 0, 2) Clase 1, 3) Clase 2 y 4) Clase 3. Los Propósitos de Prueba del Nivel DLC son comunes para la Terminación Portátil y la Terminación Fija.

Por último, los Casos de Prueba para el Nivel de Acceso al Medio están divididas en cinco grupos: 1) Servicios de Difusión, 2) Servicios No Orientados a Conexión, 3) Servicios Orientados a Conexión, 4) Procedimientos de Gestión del Nivel y 5) Procedimientos de Mensajes de Prueba (grupo vacío). Los Propósitos de Prueba del Nivel MAC son comunes para la Terminación Portátil y la Terminación Fija.

Para el Perfil GAP, la Especificación de Pruebas del Perfil (PTS) está descrita en la parte 1 del documento [ETS 300 494]. Las partes 2 y 3 del mismo documento indican la Estructura y Propósito de las Pruebas del Perfil GAP para la Terminación Portátil y la Terminación Fija, respectivamente. En el caso de los protocolos no existen Propósitos de Prueba adicionales a los definidos para la Interfaz Común. El documento [TBR 022] incluye, de forma resumida, las funcionalidades aplicables a GAP junto con la lista de Casos de Prueba que son de aplicación.

Adicionalmente a las pruebas de conformidad descritas en los anteriores documentos, el documento [I-ETS 300 176] es de aplicación en todos aquellos aspectos relacionados con el uso del espectro de frecuencias (parte 1) y la codificación de voz según el estándar ADPCM (parte 2) (*Adaptive Differential Pulse-Code Modulation*) [G.726]; son pruebas para evitar que un equipo no dañe o provoque perjuicios a otros equipos o servicios.

8.3 Definición del Sistema de Pruebas

En esta fase se decide que los Sistemas de Pruebas a implementar serán los siguientes:

- Sistemas de Pruebas para el Nivel DLC:
  - Terminación PT (`SP_DLC_PT`).
  - Terminación FT (`SP_DLC_FT`).
- Sistema de Pruebas para el Nivel NWK:
  - Terminación PT (`SP_NWK_PT`).

Los Juegos de Pruebas Abstractas sólo admiten el Método de Pruebas remoto empotrado de una sola capa (Sección 8.4.1.1). El conjunto de Casos de Prueba Abstractos que se ha incluido en cada Sistema de Pruebas ha estado restringido por la funcionalidad del Módulo de Capa Física. Este módulo, como se verá más adelante (Sección 8.5.1.1.1), se encuentra implementado en unos módulos comerciales que poseen las siguientes limitaciones:

- No trabajan con identificadores SARI (*Secondary Access Rights Identity*).
- No es posible provocar un traspaso.
- No incluyen los algoritmos de cifrado.[9]

[9] Los algoritmos de cifrado del Sistema DECT están disponibles sólo bajo el pago de una licencia y no se incluyen en las placas que se han utilizado. Este hecho afecta a varios Casos de Prueba del Nivel de Red.

SPICUM servicio de publicaciones

- No disponen de primitivas de arranque y parada del cifrado.

La Selección de Casos de Prueba[10] está formada por todos los Casos de Prueba relativos al Perfil GAP menos los afectados por estas limitaciones (ver Apéndice H). La Tabla 8.3 muestra el total de los Grupos y Casos de Prueba disponibles en GAP e implementados en cada Sistema de Pruebas. Las Selecciones de Casos de Prueba han sido guardadas en los archivos:

- `SP_DLC_PT_Gap.itex`: para el Sistema de Pruebas DLC-PT.
- `SP_DLC_FT_Gap.itex`: para el Sistema de Pruebas DLC-FT.
- `SP_NWK_PT_Gap.itex`: para el Sistema de Pruebas NWK-PT.

Tabla 8.3: Resumen de los Casos de Prueba incluidos en los Sistemas de Pruebas construidos.

Nivel	Método de Pruebas	Terminación	GAP		Sistemas de Pruebas	
			Grupos de Pruebas	Casos de Prueba	Grupos de Pruebas	Casos de Prueba
DLC	Remoto empotrado de una sola capa	PT	11	30	11	28
		FT	10	22	10	20
NWK		PT	31	114	29	78

8.4 Construcción del Subsistema de Pruebas

A continuación se describe la construcción de los Subsistemas de Pruebas de cada Sistema de Pruebas. En primer lugar se describen los Juegos de Pruebas utilizados, incluyendo los Métodos de Pruebas definidos; se diferencia entre pruebas de conformidad de protocolo y pruebas de conformidad de perfiles. A continuación, se explica cómo se ha obtenido la entidad ejecutable de este Subsistema.

8.4.1 Construcción de los Juegos de Pruebas Abstractas

Los Juegos de Pruebas Abstractas (ATS) para DECT han sido modelados por ETSI. Por ello, esta fase se limita a recopilar los módulos y documentos aplicables a cada Subsistema de Pruebas. En primer lugar se indica la documentación relacionada con las Pruebas de las Especificaciones de Base y posteriormente la relacionada con el perfil GAP. Cada apartado se ha dividido en subsecciones (Método de Pruebas, Juego de Pruebas Abstractas, Declaraciones de Conformidad e Informes de Pruebas) de forma similar a como se ha estructurado la descripción de la documentación en la Sección 2.2.5 del Capítulo 2.

8.4.1.1 Pruebas de Conformidad de Protocolo

En las Pruebas de Conformidad de protocolo de DECT cada nivel se certifica de forma independiente de los demás. Los distintos elementos se encuentran recogidos en las normas [EN 300 497-5] (DLC) y [EN 300 497-7] (NWK-PT). La lista completa de documentos se muestra en la Tabla 8.4.

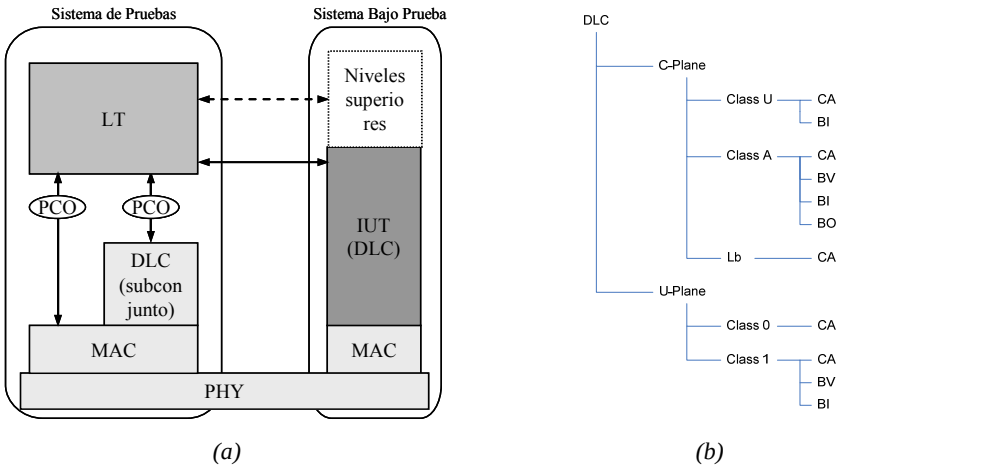
[10] A partir de ahora, por simplicidad, se utilizará el nombre de Juegos de Pruebas Abstractas también para denotar a esta Selección de Casos de Prueba.

SPICUM servicio de publicaciones

Tabla 8.4: Especificaciones de Prueba aplicables.

Documento	Título
Formularios de Declaración de Conformidad de Implementación de Protocolo	
ETS 300 476-1	Digital Enhanced Cordless Telecommunications (DECT); Common Interface (CI); Protocol Implementation Conformance Statement (PICS) proforma; Part 1: Network (NWK) layer – Portable radio Termination (PT)
ETS 300 476-2	Digital Enhanced Cordless Telecommunications (DECT); Common Interface (CI); Protocol Implementation Conformance Statement (PICS) proforma; Part 2: Data Link Control (DLC) layer - Portable radio Termination (PT)
ETS 300 476-5	Digital Enhanced Cordless Telecommunications (DECT); Common Interface (CI); Protocol Implementation Conformance Statement (PICS) proforma; Part 5: Data Link Control (DLC) layer - Fixed radio Termination (FT)
Métodos de Pruebas, Juegos de Pruebas, Formularios de Declaración de Información Suplementaria para Pruebas e Informes de Pruebas de Conformidad de Protocolo	
EN 300 497-5	Digital Enhanced Cordless Telecommunications (DECT); Common Interface (CI); Test Case Library (TCL); Part 5: Abstract Test Suite (ATS) - Data Link Control (DLC) layer
EN 300 497-7	Digital Enhanced Cordless Telecommunications (DECT); Common Interface (CI); Test Case Library (TCL); Part 7: Abstract Test Suite (ATS) for Network (NWK) layer - Portable radio Termination (PT)

Se emplea el Método de Pruebas remoto de una sola capa en la variante empotrada. El Método de Pruebas para el Nivel DLC (Figura 8.4-a) posee dos PCOs que comunican el Comprobador Inferior con el Nivel MAC y un subconjunto de la funcionalidad del DLC, ambos incluidos en el Subsistema Inferior. Este subconjunto es el encargado de realizar la fragmentación y la recombinación de las primitivas de datos (`MAC-CO-DATA`). Por su parte, el Método de Pruebas para el Nivel NWK (Figura 8.5-a) posee un único PCO, que actúa como interfaz entre el Comprobador Inferior y el Nivel DLC del Subsistema Inferior.



(a) *(b)*

Figura 8.4: (a) Método de Pruebas y (b) Grupos de Pruebas para el Nivel DLC (FT/PT).

Los Juegos de Pruebas Abstractas están escritos en notación TTCN. La Tabla 8.3 muestra el volumen de Pruebas para cada nivel. La Figura 8.4-b y la Figura 8.5-b muestran, respectivamente, un esquema de los Grupos de Pruebas de DLC y NWK-PT. La lista completa de los Grupos y Casos de Prueba se encuentra en el Apéndice H.

SPICUM servicio de publicaciones

Los formularios de Declaración de Conformidad de Implementación de Protocolo (PICS están incluidos en el documento [ETS 300 476], partes 1 (NWK-PT), 2 (DLC-PT) y 5 (DLC-FT). Los formularios de las Declaraciones de Información Suplementaria de Implementación para Pruebas (PIXIT) aparecen en el mismo documento que los Juegos de Pruebas Abstractas (Tabla 8.4). Para una Especificación Base no existe el formulario de Declaración de Conformidad de Sistema (SCS) [ETS 300 406].[11]

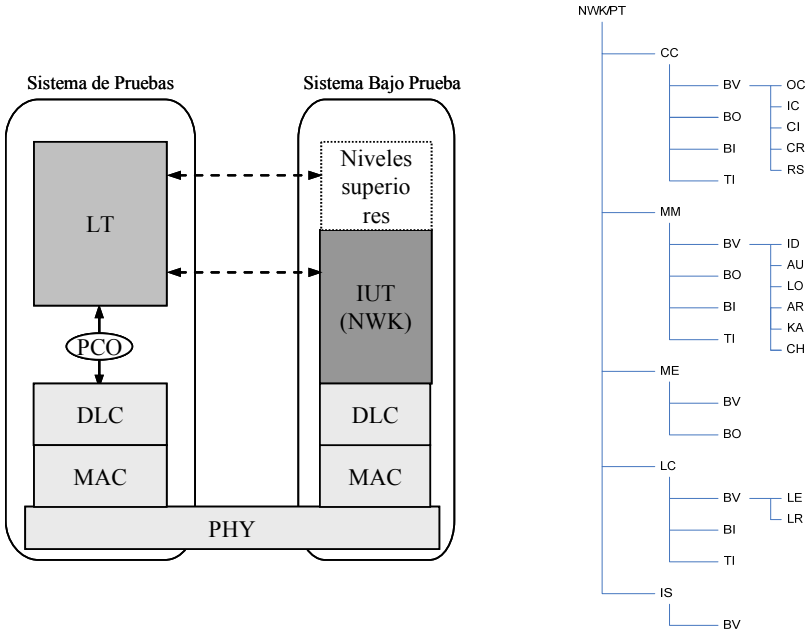


Figura 8.5: (a) Método de Pruebas y (b) Grupos de Pruebas para el Nivel NWK (PT).

Los formularios de Informe de Pruebas de Conformidad de Protocolo (PCTR) están incluidos en el mismo documento que los Juegos de Pruebas (Tabla 8.4). Para una Especificación Base no existe el formulario de Informe de Pruebas de Conformidad de Sistema (SCTR) [ETS 300 406].

8.4.1.2 Pruebas de Conformidad de Perfiles

Las Pruebas de Conformidad del perfil GAP están basadas en las Pruebas de las Especificaciones Base. La Especificación de las Pruebas Específicas del Perfil (PSTS) está descrita en los documentos [ETS 300 494-2], para la PT, y [ETS 300 494-3], para la FT.

Los Métodos de Pruebas Abstractas y los Juegos de Pruebas Abstractas son los mismos que para las Especificaciones Base. La Especificación de las Pruebas Específicas del Perfil incluye una lista de los Grupos y Casos de Prueba aplicables a una implementación GAP.

Los documentos [ETS 300 474-1] (PT) y [ETS 300 474-2] (FT), incluye modificaciones de los formularios de Declaración de Conformidad de Implementación de Protocolo (PICS) y un formulario adicional de Declaración Específica de Conformidad de

[11] Aunque según la Metodología de Pruebas de Conformidad OSI este documento sí es pertinente para las Pruebas de Especificaciones Base, ETSI considera que sólo es relevante para los perfiles.

SPICUM servicio de publicaciones

Implementación de Perfil. Los formularios de Declaración de Información Suplementaria de Implementación para Pruebas de Protocolo aplicables son los mismos que para las Especificaciones Base. Los formularios de Declaración de Conformidad de Sistema (SCS) están incluidos en la Especificación de las Pruebas Específicas del Perfil.

Los formularios de Informe de Pruebas de Conformidad de Protocolo (PCTR) están incluidos en la Especificación de las Pruebas Específicas del Perfil. Los formularios de Informe de Pruebas de Conformidad de Sistema (SCTR) están incluidos en la Especificación de las Pruebas Específicas del Perfil.

Tabla 8.5: Documentos resultantes de la fase Documentación.

Documento	Título
Formularios de Declaración de Conformidad de Implementación de Protocolo	
ETS 300 474-1	Digital Enhanced Cordless Telecommunications (DECT); Generic Access Profile (GAP); Profile requirement list and profile specific Implementation Conformance Statement (ICS) proforma; Part 1: Portable radio Termination (PT)
ETS 300 474-2	Digital Enhanced Cordless Telecommunications (DECT); Generic Access Profile (GAP); Profile requirement list and profile specific Implementation Conformance Statement (ICS) proforma; Part 2: Fixed radio Termination (FT)
Especificación de las Pruebas Específicas de Perfil	
ETS 300 494-2	Digital Enhanced Cordless Telecommunications (DECT); Generic Access Profile (GAP); Profile Test Specification (PTS); Part 2: Profile Specific Test Specification (PSTS) - Portable radio Termination (PT)
ETS 300 494-3	Digital Enhanced Cordless Telecommunications (DECT); Generic Access Profile (GAP); Profile Test Specification (PTS); Part 3: Profile Specific Test Specification (PSTS) - Fixed radio Termination (FT)

8.4.2 Diseño del Juego de Pruebas Ejecutables

Las tareas que se realizan en esta fase están resumidas en la Figura 8.6. Los Juegos de Pruebas Ejecutables (ETS) se generan a partir de los Juegos de Pruebas Abstractas elaborados en la etapa anterior. Ha sido necesario efectuar algunos retoques en estos ficheros ya que algunos aspectos no estaban definidos de acuerdo con las Especificaciones de Sistema. Aun cuando la norma [ETS 300 175-4] define algunas primitivas con campos opcionales, o incluso omitidos, en el código TTCN se obligaba la presencia de esos campos. Las modificaciones realizadas han sido cambiar en estos casos la especificación de recepción '`?`' (espera recibir cualquier cosa) por un comodín '`*`' (espera recibir cualquier cosa, o ninguna).

La herramienta de generación de código de Tau Suite (Capítulo 5, Sección 5.1) produce el conjunto de ficheros mostrado en la Tabla 8.6 a partir de los Casos de Prueba seleccionados. Por un problema de la herramienta de generación de código, algunas inicializaciones de variables se realizaban posteriormente a su utilización, por lo que ha sido necesario modificar el fichero `s_decl.c` y cambiar el orden de las líneas de código adecuadamente.

Los ficheros de Parámetros de Pruebas y las declaraciones de las funciones TSO se han generado mediante la herramienta *GenDef* a partir de los Juegos de Pruebas Abstractas (ficheros `.mp`). La lista de las funciones TSO utilizadas en los Juegos de Pruebas, junto con una breve descripción de las mismas, aparece, para cada Sistema de Pruebas, en la Tabla 8.8 y la Tabla 8.9.

SPICUM servicio de publicaciones

Algunas de estas funciones requieren el procesamiento de los parámetros (`GcTSO_cid_checksum`) o la conversión de los contenidos a un formato diferente (`GcTSO_cinpt_int_to_oct_1`). Otras invocan acciones concretas, como puede ser el activar o desactivar determinada funcionalidad (`GcTSO_cinpt_set_bit_a38`). Un último grupo requiere alguna actuación por parte del operador, que puede ser realizar una acción (`GcTSO_lose_and_regain_lock`) o comprobar el funcionamiento de algún aspecto de la IUT (`GcTSO_check_stand_char`).

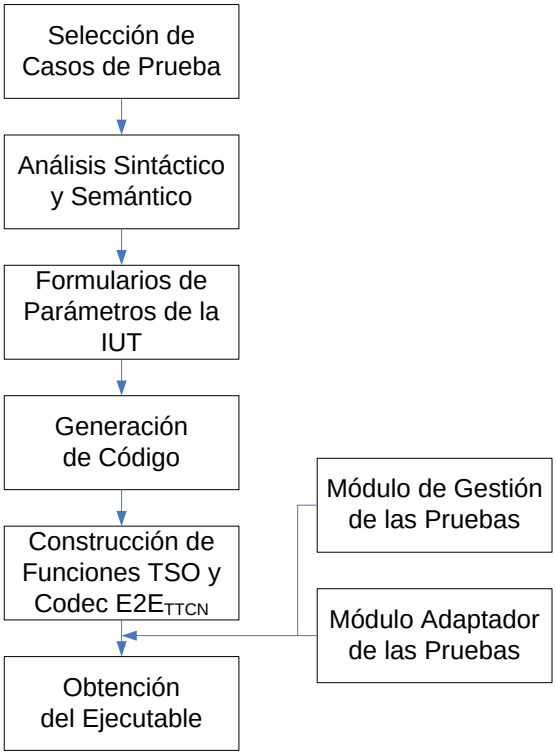


Figura 8.6: Tareas para la generación del ejecutable del Subsistema de Pruebas.

Se ha modificado el fichero `s_constr.c` porque la declaración de la TSO de cálculo de la suma de comprobación para una trama que se envía (`GcTSO_cid_checksum`), está definida de manera que no contiene parámetros de entrada. Sin embargo, debe poder acceder a la trama en curso para realizar el cálculo. Al no ver forma de acceder a esta información mediante una variable global, ha sido necesario facilitarle la variable que contenía la trama.

Tabla 8.6: Lista de ficheros producidos por el generador de código C a partir de módulos TTCN.

`gcihard.c`	`gcihard.h`	`s_aux.c`	`s_aux.h`
`s_constr.c`	`s_constr.h`	`s_decl.c`	`s_decl.h`
`s_dyn.c`	`s_dyn.h`	`s_type.c`	`s_type.h`

El código generado a partir de los Juegos de Pruebas, junto con el código implementado de las funciones TSO, se enlaza con los Módulos Adaptador y de Gestión de las Pruebas

SPICUM servicio de publicaciones

y con las librerías del Motor TTCN para obtener el ejecutable del Subsistema de Pruebas.

Los ficheros correspondientes a cada Sistema de Pruebas se muestran en la Tabla 8.7.

Tabla 8.7: Ficheros de Pruebas Ejecutables y de Parámetros de Pruebas para cada Sistema de Pruebas.

	Entradas	Salidas	
Sistema de Pruebas	**Juegos de Pruebas Abstractas**	**Subsistema de Pruebas**	**Parámetros de Pruebas**
SP_DLC_PT	SP_DLC_PT_Gap.mp	Ets_dlc_pt.exe	pixit_dlc_pt.ttp
SP_DLC_FT	SP_DLC_FT_Gap.mp	Ets_dlc_ft.exe	pixit_dlc_ft.ttp
SP_NWK_PT	SP_NWK_PT_Gap.mp	Ets_nwk_pt.exe	pixit_nwk_pt.ttp

Tabla 8.8: Funciones TSO de los Sistemas de Pruebas SP_DLC_PT (tso_dlc_pt.c) y SP_DLC_FT (tso_dlc_ft.c).

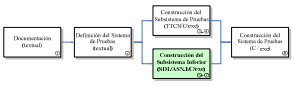
Función	**Resultado**	**Descripción**	**SP_DLC_XX**	
			PT	**FT**
GcTSO_cid_checksum (VcValue* data)	OCTETSTRING	Calcula la suma de comprobación de los datos enviados.	✓	✓
GcTSO_check_checksum ()	OCTETSTRING	Comprueba la suma de comprobación de los datos recibidos.	✓	✓
GcTSO_compute_li (VcValue * nwkmsg)	INTEGER	Calcula el tamaño de un mensaje de nivel de red.	✓	✓
GcTSO_cid_fill (VcValue * chn, VcValue * length)	FILLSTRING	Devuelve los octetos de relleno necesarios para que la longitud sea múltiplo de 8.	✓	✓
GcTSO_cid_lowest (VcValue * nb, VcValue * string)	BITSTRING	Devuelve los nb bits menos significativos de la cadena string.	✓	✓
GcTSO_cid_return_cr_value (VcValue * iut_type, VcValue * frame_type, VcValue * send_constr)	INTEGER	Devuelve el valor del bit CR.	✓	✓
GcTSO_set_bit_a38_to_0 ()	BOOLEAN	Desactiva el bit a38 del canal de difusión (área de localización).	-	✓
GcTSO_ignore_bearer_handover (VcValue * rpn)	BOOLEAN	Se indica a la PT que no están permitidos los traspasos. Se ignoran las peticiones.	-	✓
GcTSO_inhibit_intracell_handover ()	BOOLEAN	Inhibe la realización de traspasos en la misma celda.	-	✓
GcTSO_cho_blind_slot_info ()	BOOLEAN	Transmite información de intervalo ciego en la posición de las portadoras piloto para evitar que la IUT las encuentre.	-	✓

SPICUM servicio de publicaciones

Tabla 8.9: Funciones TSO del Sistema de Pruebas `SP_NWK_PT` (`tso_nwk_pt.c`).

Función	Resultado	Descripción
`GcTSO_cinpt_bitstr_inc (VcValue * bitstr)`	`BITSTRING`	Suma 1 a un entero representado como un `BITSTRING`.
`GcTSO_cinpt_calculate_w_from_TPUI (VcValue * tpui)`	`BIT_1`	Indica si la dirección TPUI (*Temporary Portable User Identity*) proviene de la TPUI asignada o de la TPUI por defecto.
`GcTSO_check_calling_party_number_ind ()`	`BOOLEAN`	Comprueba que el número llamante se muestra correctamente.
`GcTSO_check_ctrl_char ()`	`BOOLEAN`	Comprueba si la IUT entiende correctamente los caracteres de control; para ello, borra la pantalla.
`GcTSO_cinpt_check_link_present ()`	`BOOLEAN`	Comprueba si se encuentra presente un enlace de nivel 2.
`GcTSO_cinpt_change_location_area ()`	`BOOLEAN`	Establece un nuevo piloto en la misma celda, pero en otra RFP y reduce la potencia de la RFP inicial para que la IUT se enganche a la nueva.
`GcTSO_cinpt_lowest (VcValue * nb, VcValue * string)`	`BITSTRING`	Devuelve los `nb` bits menos significativos de la cadena `string`.
`GcTSO_cinpt_set_bit_a38 (VcValue * param)`	`BOOLEAN`	Pone el bit `a38` al valor indicado por `param`.
`GcTSO_check_stand_char ()`	`BOOLEAN`	Comprueba la impresión de caracteres estándar en pantalla; muestra la cadena `1234`.
`GcTSO_cinpt_check_u_plane (VcValue * dlei)`	`BOOLEAN`	Comprueba la conexión del plano U indicada por indicada por `dlei`.
`GcTSO_check_user_alerting ()`	`BOOLEAN`	Pregunta si se ha oído el timbre de aviso de llamada.
`GcTSO_cinpt_convert_ac_to_bitstring (VcValue * param)`	`BIT_32`	Transforma la representación en dos octetos (BCD – *Binary-Coded Decimal*) de los caracteres de la cadena `param` a una representación de 32 bits.
`GcTSO_cinpt_convert_upi_to_bitstring (VcValue * param)`	`BIT_32`	Convierte el valor decimal `param` representado en BCD a una representación de 32 bits.
`GcTSO_fill_cinpt_portable_id_ipui (VcValue * value, VcValue * length)`	`PORT_ID_VALUE_TYPE`	Rellena el campo `id_value` del elemento de información `PORTABLE_ID` según el tipo de IPUI (*International Portable User Identity*).
`GcTSO_fill_cinpt_portable_id_length (VcValue * param)`	`OCT_1`	Calcula la longitud real elemento de información `PORTABLE_ID` pasado como parámetro.
`GcTSO_init_broadcast_bits ()`	`BOOLEAN`	Activa los bits `a38` (área de localización) y `a44` (capacidades de los niveles superiores) del canal de difusión.
`GcTSO_cinpt_int_to_oct_1 (VcValue * param)`	`OCT_1`	Convierte un valor entero (menor de 255) en dos octetos en BCD.
`GcTSO_lose_and_regain_lock ()`	`BOOLEAN`	Solicita que se apague y se encienda la IUT.
`GcTSO_set_bit_a44 (VcValue * param)`	`BOOLEAN`	Pone el bit 'a44' al valor indicado.
`GcTSO_oct_to_int (VcValue * octetstring)`	`INTEGER`	Convierte el octeto `octetstring` a entero.
`GcTSO_print (VcValue * string)`	`BOOLEAN`	Presenta un mensaje por pantalla y solicita la confirmación del operador.
`GcTSO_concat (VcValue * Ostring1, VcValue * Ostring2)`	`DECT_1_255`	Concatena `Ostring1` y `Ostring2`.
`GcTSO_use_PARK_as_PARI (VcValue * park)`	`BOOLEAN`	Fija como PARI (*Primary Access Rights Identity*) del Sistema de Pruebas el valor de PARK (*Portable Access Right Key*) indicado en el parámetro.
`GcTSO_disable_connection_handover_fp1 ()`	`BOOLEAN`	Inhabilita el traspaso de conexión para la Terminación Fija FP1.
`GcTSO_cho_blind_slot_info ()`	`BOOLEAN`	Transmite información de intervalo ciego en la posición de las portadoras piloto para evitar que la IUT las encuentre.

SPICUM servicio de publicaciones



8.5 Construcción del Subsistema Inferior

En los siguientes apartados se describen las actividades realizadas para construir el Subsistema Inferior de cada Sistema de Pruebas (Figura 8.7).

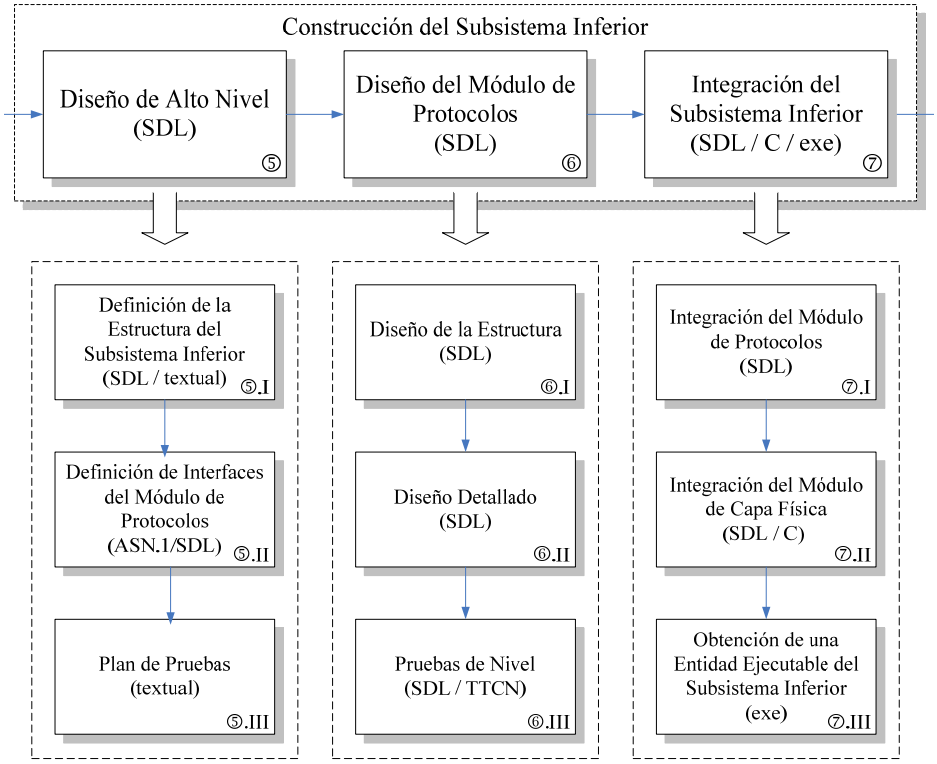


Figura 8.7: Fases para la construcción del Subsistema Inferior.

8.5.1 Diseño de Alto Nivel

En la fase de Diseño de Alto Nivel se define la estructura interna del Subsistema Inferior, separando funcionalidades entre el Módulo de Protocolos y el Módulo de Capa Física. Asimismo, se describen las interfaces que el Módulo de Protocolos utilizará, tanto las interfaces externas (Subsistema de Pruebas, Módulo de Capa Física) como las interfaces internas entre sus bloques. Finalmente, se presenta el Plan de Pruebas.

8.5.1.1 Definición de la Estructura del Subsistema Inferior

El Subsistema Inferior está formado por el Módulo de Protocolos y el Módulo de Capa Física (Capítulo 3, Figura 3.10). Seguidamente se describe su estructura.

8.5.1.1.1 División funcional entre el Módulo de Protocolos y el Módulo de Capa Física

La separación entre ambos Módulos se ha realizado en el Nivel de Acceso al Medio, ya que para realizar el Módulo de Capa Física se ha optado por utilizar unos módulos de la

SPICUM servicio de publicaciones

compañía SiTel Sierra[12] ([SIT94a], [SIT94b]). Estos módulos implementan el Nivel Físico y la parte inferior del Nivel MAC (CSF – Funciones de Celda del Emplazamiento (*Cell Site Functions*)). El Módulo de Protocolos será el encargado de implementar el resto de la funcionalidad del Subsistema Inferior. La Tabla 8.10 muestra los módulos que pertenecen a cada Subsistema de Pruebas.

Tabla 8.10: Módulos que constituyen el Subsistema Inferior de cada Sistema de Pruebas.

	Sistemas de Pruebas		
Módulo	**SP_DLC_PT**	**SP_DLC_FT**	**SP_NWK_PT**
Módulo de Protocolos	*MProt_DLC_PT*	*MProt_DLC_FT*	*MProt_NWK_PT*
Módulo de Capa Física	*MFis_FT*	*MFis_PT*	*MFis_FT*

Módulo de Capa Física

Este módulo integra el Nivel Físico y las funciones inferiores (CSF) del Nivel MAC (Tabla 8.12). Hay un módulo distinto según si se desea operar en el rol de Terminación Portátil o Terminación Fija; su control se realiza a través de una línea serie RS-232.

La entidad CSF proporciona controladores de difusión (DBC – *Dummy Bearer Control*), que gestionan el envío de los canales N, Q y P en la Terminación Fija, y de tráfico (TBC – *Traffic Bearer Control*), que gestionan los canales de datos. Soporta un máximo de cuatro portadoras de tráfico para la RFP (*Radio Fixed Part*) y dos para la PT. La separación entre ellas debe seguir la regla: N, N + 2, N + 5; siendo los intervalos 10 y 11 no utilizables. De esta forma, la RFP puede manejar 2 llamadas locales y una externa simultáneamente y es capaz de realizar un traspaso de canal en cada momento. La PT puede soportar una llamada y el procedimiento de traspaso. La Tabla 8.11 resume los servicios soportados.

Esta funcionalidad no corresponde a la totalidad de la funcionalidad requerida para el Perfil de Acceso Genérico; estas limitaciones tienen su impacto en el conjunto de Pruebas implementables en los Sistemas de Pruebas, como se indicó en la sección 8.3. Los servicios afectados son los siguientes:

- Difusión de mensajes de aviso: Se admiten mensajes de aviso de tipo corto y largo (*short-page* o *full-page*), pero no mensajes de tamaño cero (*zero-page*), que sólo llevan información de nivel MAC. El Perfil GAP requiere mensajes de tipo corto y de tamaño cero.
- Activación/Desactivación del cifrado: No se proporciona ninguna interfaz para enviar los mensajes del protocolo de activación/desactivación del cifrado entre la PT y la FT. Al carecer de las primitivas necesarias, no es posible que la PT informe a la FT de que quiere comenzar a cifrar la información, y por tanto este servicio no se puede realizar.
- Localización de frecuencias adicionales: Sólo se puede trabajar con las portadoras preestablecidas.

[12] La compañía SiTel Sierra fue adquirida por Nacional Semiconductor en enero de 1996.

SPICUM servicio de publicaciones

- Soporte para derechos de acceso secundarios (SARI): En la PT no hay primitivas que permitan leerlos.

Tabla 8.11: Servicios proporcionados por los módulos de Sitel [SIT94c].

	Servicios
Canales físicos	• 10 portadoras RF (*Radio Frequency*) con un solo transceptor. • Portadoras full duplex y simplex. • Traspaso de canal físico.
Canales lógicos	• Transmisión de los canales lógicos Cs e I_N. • Transmisión del canal de aviso P entre RFP y PT. • Transmisión del canal Q para sincronización y envío de información del sistema.
Control de calidad	• Compensación del nivel de continua y ajuste de fase. • Lectura del nivel de señal (RSSI - *Received Signal Strength Indication*) tanto en la RFP como en la PT. • Aleatorización de bits. • Desactivación del canal de voz.

8.5.1.1.2 Definición de la estructura del Módulo de Protocolos

La estructura de cada Módulo de Protocolos se muestra en la Figura 8.8 y la Figura 8.9. Cada subcomponente de este módulo se ha modelado como un bloque SDL; el nombre asignado a cada bloque contiene una palabra significativa de la funcionalidad o nivel que implementa, con un sufijo dependiente del rol en el que actúa. La Tabla 8.12 lista los subcomponentes incluidos en cada módulo.

Tabla 8.12: Subcomponentes de cada Módulo del Subsistema Inferior para cada Sistema de Pruebas.

		Sistemas de Pruebas		
Módulo		**SP_DLC_PT**	**SP_DLC_FT**	**SP_NWK_PT**
Módulo de Protocolos	*Nombre*	*MProt_DLC_PT*	*MProt_DLC_FT*	*MProt_NWK_PT*
	Bloques	LLME_MAC_FT SUB_DLC_FT MAC_CCF_FT LINSER_FT	LLME_MAC_PT SUB_DLC_PT MAC_CCF_PT LINSER_PT	LLME_FT DLC_FT MAC_CCF_FT LINSER_FT AJUSTE_TIPOS_FT
Módulo de Capa Física	*Nombre*	*MFis_FT*	*MFis_PT*	*MFis_FT*
	Bloques	MAC_CSF_FT PHY_FT	MAC_CSF_PT PHY_PT	MAC_CSF_FT PHY_FT

8.5.1.1.3 Asignación de responsabilidades a los subcomponentes

La funcionalidad que debe ofrecer el Módulo de Protocolos corresponde a la definida en las Especificaciones Base y las Especificaciones de Perfil GAP para cada uno de los subcomponentes.

SPICUM servicio de publicaciones

Los bloques incluidos en los Sistemas de Pruebas para el Nivel DLC (Figura 8.8) realizan las siguientes funciones:

- `MAC_CCF_PT/FT`: Modela el comportamiento del Nivel MAC según [ETS 300 175-3]. Además, implementa el comportamiento esperado por las señales `TTCN_MAC_XXX` descritas en la Sección 8.5.1.2.4 (interfaz adicional de control).
- `SUB_DLC_PT/FT`: Realiza aquella funcionalidad del Nivel DLC que las Pruebas no implementan. En concreto, asume la responsabilidad de fragmentar y recombinar las tramas DLC.

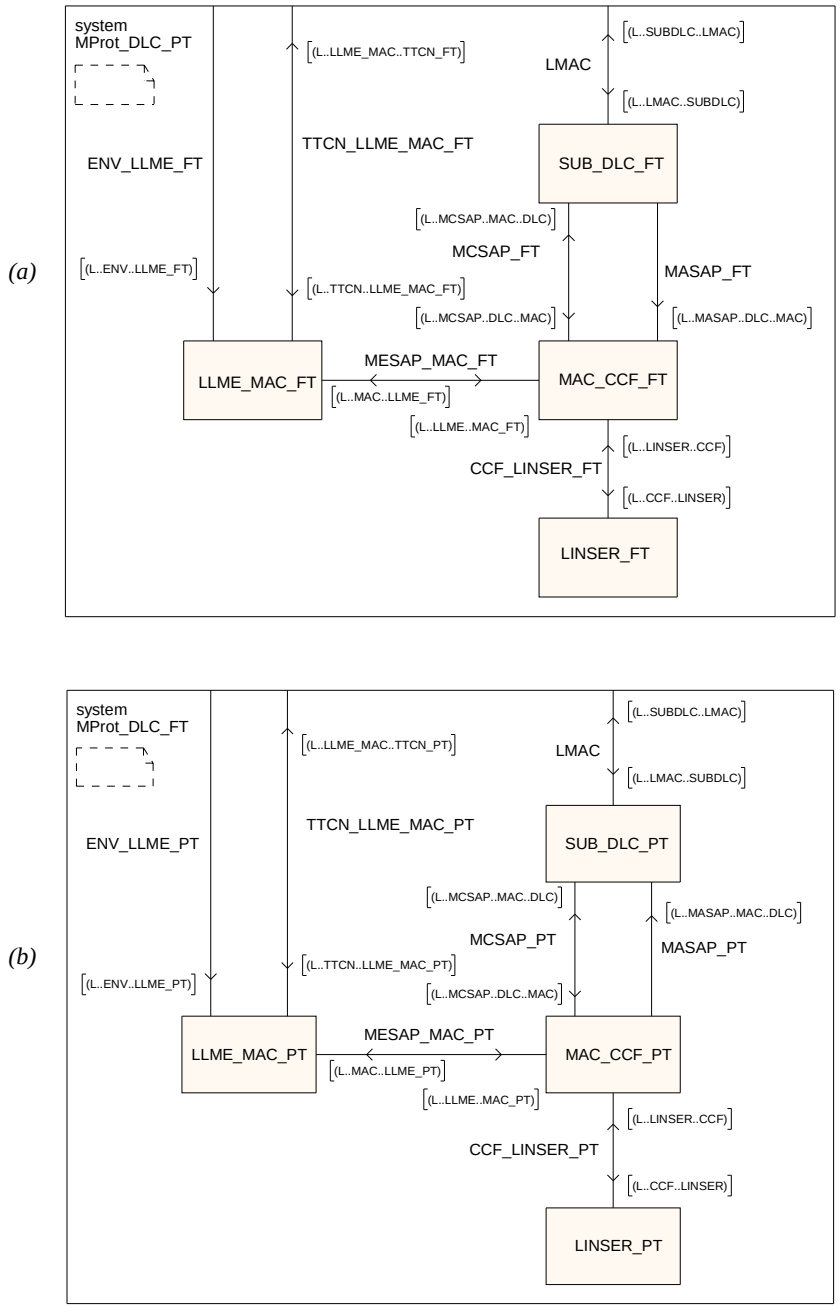


Figura 8.8: Estructura del Módulo de Protocolos para los Sistemas de Pruebas del Nivel DLC de la Terminación (a) Portátil y (b) Fija.

SPICUM servicio de publicaciones

- `LLME_MAC_PT/FT`: Contiene la funcionalidad de gestión del Nivel MAC según se define en [ETS 300 175-3]. Entre otras cosas, almacena la información de configuración del Nivel y se encarga de reinicializar el sistema y configurar los derechos y capacidades de la Terminación (interfaz adicional de control).
- `LINSER_PT/FT`: Es el encargado de comunicar el Módulo de Protocolos con el Módulo de Capa Física. Controla el puerto serie que constituye dicha interfaz y codifica/descodifica la información entre la representación abstracta en SDL y su representación en la línea serie.

Los bloques incluidos en el Sistema de Pruebas para el Nivel NWK (Figura 8.9) realizan las siguientes funciones:

- `MAC_CCF_FT`: Modela el comportamiento del Nivel MAC según [ETS 300 175-3].
- `DLC_FT`: Modela el comportamiento del Nivel DLC según [ETS 300 175-4].
- `LLME_FT`: Engloba la funcionalidad de gestión de los Niveles MAC y DLC según se define en [ETS 300 175-3] y [ETS 300 175-4]. Entre otras cosas, almacena la información de configuración de los Niveles. La funcionalidad de gestión de los Niveles MAC y DLC se ha agrupado en un solo subcomponente.
- `LINSER_FT`: Tiene la misma funcionalidad que en los Sistemas de Pruebas para el Nivel DLC.

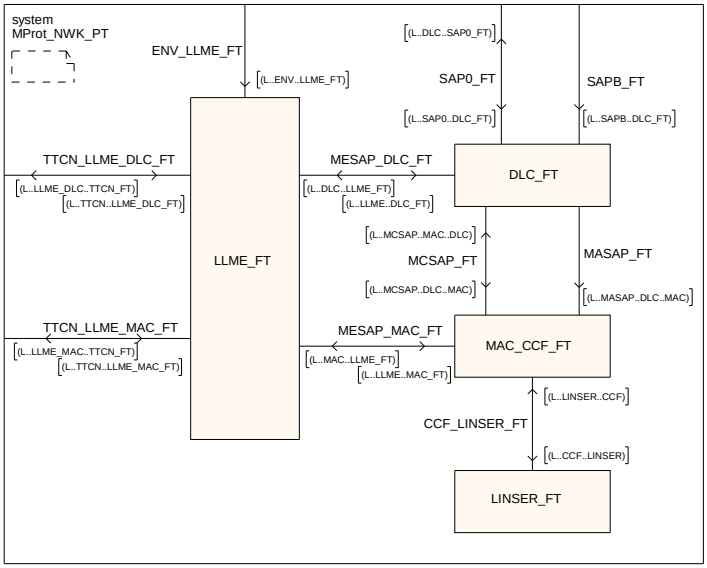


Figura 8.9: Estructura del Módulo de Protocolos para el Sistema de Pruebas del Nivel NWK de la Terminación Portátil

Además, el procesamiento de las señales de la interfaz adicional de control (Sección 8.5.1.2.4) se realiza en los siguientes bloques:

- Sistema de Pruebas para el Nivel DLC:
  - `MAC_CCF_PT/FT`: Responsable de responder a las señales `TTCN_MAC_XXX`.
  - `LLME_MAC_PT/FT`: Reinicializa el sistema y configura los derechos y capacidades de la Terminación.

SPICUM servicio de publicaciones

- Sistema de Pruebas para el Nivel NWK:
  - `LLME_FT`: Almacena la información de configuración de los Niveles.

8.5.1.2 Definición de Interfaces del Módulo de Protocolos

La Tabla 8.13 resume todas las interfaces mostradas en la Figura 8.8 y la Figura 8.9 indicando, para cada canal, las listas de señales que transporta y la interfaz a que pertenece (Capítulo 4, Sección 4.5.1.2). Las columnas tercera y cuarta señalan el sentido en que se transportan las señales de la lista correspondiente. Por convenio, se ha asumido que el símbolo '↑' indica que las señales son ascendentes en la torre de protocolos; igualmente, el símbolo '←' se ha considerado representativo de señales que son transportadas horizontalmente (dentro del mismo nivel), y hacia la izquierda en los diagramas SDL. Los símbolos '↓' y '→' representan, respectivamente, señales que son descendentes y señales que son transportadas horizontalmente hacia la derecha. No se ha incluido la descripción de los tipos de datos de los parámetros transportados por las señales por limitar el detalle descriptivo de esta sección.

Tabla 8.13: Lista de interfaces del diseño de alto nivel en cada Sistema de Pruebas.

Sistema de Pruebas	Canal	↑ / ←	↓ / →	Interfaz[1]
SP_DLC_PT	LMAC	L..SUBDLC..LMAC	L..LMAC..SUBDLC	SubPru
	MCSAP_FT	L..MCSAP..MAC..DLC	L..MCSAP..DLC..MAC	Interno
	MASAP_FT	---	L..MASAP..DLC..MAC	Interno
	MESAP_MAC_FT	L..MAC..LLME_FT	L..LLME..MAC_FT	Interno
	CCF_LINSER_FT	L..LINSER..CCF	L..CCF..LINSER	Interno
	TTCN_LLME_MAC_FT	L..LLME_MAC..TTCN_FT	L..TTCN..LLME_MAC_FT	Control-TSOs
	ENV_LLME_FT	---	L..ENV..LLME_FT	Control-Pruebas
SP_DLC_FT	LMAC	L..SUBDLC..LMAC	L..LMAC..SUBDLC	SubPru
	MCSAP_PT	L..MCSAP..MAC..DLC	L..MCSAP..DLC..MAC	Interno
	MASAP_PT	L..MASAP..MAC..DLC	---	Interno
	MESAP_MAC_PT	L..MAC..LLME_PT	L..LLME..MAC_PT	Interno
	CCF_LINSER_PT	L..LINSER..CCF	L..CCF..LINSER	Interno
	TTCN_LLME_MAC_PT	L..LLME_MAC..TTCN_PT	L..TTCN..LLME_MAC_PT	Control-TSOs
	ENV_LLME_PT	---	L..ENV..LLME_PT	Control-Pruebas
SP_NWK_PT	SAP0_FT	L..DLC..SAP0_FT	L..SAP0..DLC_FT	SubPru
	SAPB_FT	---	L..SAPB..DLC_FT	SubPru
	MCSAP_FT	L..MCSAP..MAC..DLC	L..MCSAP..DLC..MAC	Interno
	MASAP_FT	---	L..MASAP..DLC..MAC	Interno
	MESAP_DLC_FT	L..DLC..LLME_FT	L..LLME..DLC_FT	Interno
	MESAP_MAC_FT	L..MAC..LLME_FT	L..LLME..MAC_FT	Interno
	CCF_LINSER_FT	L..LINSER..CCF	L..CCF..LINSER	Interno
	TTCN_LLME_MAC_FT	L..LLME_MAC..TTCN_FT	L..TTCN..LLME_MAC_FT	Control-TSOs
	TTCN_LLME_DLC_FT	L..LLME_DLC..TTCN_FT	L..TTCN..LLME_DLC_FT	Control-TSOs
	ENV_LLME_FT	---	L..ENV..LLME_FT	Control-TSOs

(1) SubPru – Interfaz con el Subsistema de Pruebas;
Interno – Interfaz interna del Módulo de Protocolos;
Control – Interfaz adicional de Control.

SPICUM servicio de publicaciones

A continuación se describe cada una de las interfaces; cada apartado se ha estructurado en dos subapartados, uno para los Sistemas de Pruebas del Nivel DLC y otro para el Sistema de Pruebas del Nivel NWK.

8.5.1.2.1 Interfaz con el Subsistema de Pruebas

La definición de las primitivas de la interfaz con el Subsistema de Pruebas se ha obtenido aplicando la herramienta *GenDef* a los Juegos de Pruebas (DLC o NWK); igualmente se ha obtenido la definición de las PDUs y otros tipos de datos transportados en estas primitivas.

Sobre estas declaraciones generadas automáticamente, se han realizado algunas modificaciones manuales por los siguientes motivos:

- Algunos de los tipos de datos obtenidos, por ejemplo `PDU_CC_SETUP`, están declarados en los Juegos de Prueba tanto para el Nivel DLC como para el Nivel NWK. Para evitar las inconsistencias se les ha añadido un sufijo indicativo del Juego de Pruebas del que se han derivado.
- La notación TTCN no puede expresar el concepto de campo obligatorio en una estructura, ya que asume que todos los campos son opcionales. Por ello, es necesario introducir de forma manual esta semántica en las declaraciones (palabra clave `OPTIONAL` para los campos realmente opcionales). La información se ha obtenido a partir de las Especificaciones Base ([ETS 300 175-3], [ETS 300 175-4].).

A continuación se describe la interfaz de cada Sistema de Pruebas.

Sistemas de Pruebas para el Nivel DLC

La interfaz con el Subsistema de Pruebas está constituida por un solo canal, `LMAC`, que corresponde al PCO de igual nombre de los Juegos de Pruebas. Las señales (Tabla 8.14) transportadas por este canal se encuentran definidas en un paquete. Si una señal puede llevar parámetros, se ha definido de la forma `señal(asp_señal)`, siendo el parámetro la correspondiente primitiva.

Los siguientes ficheros[13] contienen las declaraciones de las ASPs, las PDUs y los tipos abstractos de datos:

- `Tip_DLC_FTyPT_Gap.asp`
- `Tip_DLC_FTyPT_Gap.pdu`
- `Tip_DLC_FTyPT_Gap.tad`

Se han empleado los mismos ficheros para el rol de Terminación Portátil y de Terminación Fija para simplificar los diseños. En total, la interfaz está formada por 12 ASPs, 8 PDUs y 52 tipos abstractos de datos.

[13] Estos ficheros incluyen las modificaciones realizadas manualmente sobre las declaraciones obtenidas tras la aplicación de la herramienta *GenDef* a los Juegos de Pruebas.

SPICUM servicio de publicaciones

Tabla 8.14: Lista de señales válidas en el canal LMAC.

Canal	Lista de Señales	Señal[14]	MProt_DLC	
			PT	FT
LMAC	L..LMAC..SUBDLC (↓)	MAC_CON_REQ	-	✓
		MAC_DATA_REQ	✓	✓
		MAC_DIS_REQ	✓	✓
		MAC_PAGE_REQ	✓	-
	L..SUBDLC..LMAC (↑)	MAC_CON_CFM	-	✓
		MAC_CON_IND	✓	-
		MAC_DATA_IND	✓	✓
		MAC_DIS_IND	✓	✓
		MAC_PAGE_IND	-	✓

Sistema de Pruebas para el Nivel NWK

La interfaz con el Subsistema de Pruebas está constituida por los canales SAP0 y SAPB, utilizados para servicios orientados a conexión y el servicio de difusión, respectivamente. Las señales se han declarado en el mismo paquete y su definición se realiza de igual forma que en el apartado anterior.

Los siguientes ficheros[13] contienen las declaraciones correspondientes de las ASPs, las PDUs y los tipos abstractos de datos:

- Tip_NWK_FTyPT_Gap.asp[15]
- Tip_NWK_FTyPT_Gap.pdu
- Tip_NWK_FTyPT_Gap.tad

En total, la interfaz está formada por 15 ASPs, 46 PDUs y 109 tipos abstractos de datos.

Tabla 8.15: Lista de señales válidas en el canal LMAC del Módulo de Protocolos MProt_NWK_PT.

Canal	Lista de Señales	Señal[16]
SAP0	L..SAP0..DLC_FT (↓)	DL_RELEASE_REQ
		DL_DATA_REQ
		DL_ENC_KEY_REQ
	L..DLC..SAP0_FT (↑)	DL_ESTABLISH_IND
		DL_RELEASE_CFM
		DL_RELEASE_IND
		DL_DATA_IND
		DL_ENCRYPT_IND
SAPB	L..SAPB..DLC_FT (↓)	DL_BROADCAST_REQ

8.5.1.2.2 Interfaces internas del Módulo de Protocolos

Las interfaces internas corresponden, en el diseño de alto nivel, a los canales que comunican los distintos bloques del módulo. Parte de las primitivas de estas interfaces se corresponden con las extraídas de los Juegos de Pruebas Abstractas, sin embargo,

[14] Una descripción detallada de cada señal puede obtenerse en el correspondiente ATS ([EN 300 497-5]).

[15] Incluye las primitivas correspondientes a ambas Terminaciones.

[16] Una descripción detallada de cada señal puede obtenerse en el correspondiente ATS ([EN 300 497-7])

SPICUM servicio de publicaciones

otras, como las primitivas de gestión, se han definido a partir de las Especificaciones de Sistema. A continuación se describen las interfaces de cada Sistema de Pruebas.

Sistemas de Pruebas para el Nivel DLC

Las interfaces internas de ambos Módulos de Protocolos están formadas por cuatro canales: `MCSAP_PT/FT`, `MASAP_PT/FT`, `CCF_LINSER_PT/FT` y `MESAP`. Los canales `MCSAP_PT/FT` y `MASAP_PT/FT` comunican el bloque `MAC_CCF_PT/FT` con el bloque `SUB_DLC_PT/FT`. El primer canal corresponde a los servicios orientados a conexión; el segundo canal corresponde al servicio de difusión. Las señales (Tabla 8.16) de estos canales son las que circulan por el canal `LMAC` (Subsistema de Pruebas), salvo las señales las señales `MAC_DATA_REQ/IND`, que no corresponden a la verdadera interfaz del Nivel MAC. Han sido sustituidas por las señales `MAC_CO_DATA_REQ/IND` y `MAC_CO_DTR_IND` [ETS 300 175-3]. Las señales `MAC_CO_XX` transportan segmentos de una trama DLC, mientras que las señales `MAC_DATA_XX` transportan tramas DLC completas. Las primitivas se han incluido en el fichero `Otras_primitivas_DLC.asp`.

El canal `CCF_LINSER_PT/FT` es la vía de comunicación del bloque `MAC_CCF_PT/FT` con el bloque `LINSER_PT/FT`. Sus señales (Tabla 8.17) se han modelado a partir de la especificación de la interfaz con el Módulo de Capa Física (Sección 8.5.1.2.3); una descripción de los parámetros que transportan se encuentra en la misma sección. Los parámetros de estas señales no se han modelado como campos de una estructura, sino como parámetros independientes.

Por último, el canal `MESAP` comunica el bloque `MAC_CCF` con el bloque de gestión del Nivel (`LLME_MAC_FT/PT`). Las señales (Tabla 8.18) de este canal se han definido a partir de [ETS 300 175-3], que proporciona la lista de señales de gestión y la funcionalidad esperada de cada una, pero no especifica sus parámetros ni su tipo. Las primitivas se han incluido en `Otras_primitivas_MAC.asp`; los tipos de datos asociados se han declarado en `Otros_tipos_MAC.tad`.

Tabla 8.16: Listas de señales de las interfaces internas para los Sistemas de Pruebas del Nivel DLC entre el bloque MAC_CCF y el bloque SUB_DLC.

Canal	Lista de Señales	Señal[17]	MProt_DLC	
			PT	FT
MCSAP_PT MCSAP_FT	L..MCSAP..DLC..MAC (↓)	MAC_CO_DATA_REQ	✓	✓
		MAC_CON_REQ	-	✓
		MAC_DIS_REQ	✓	✓
	L..MCSAP..MAC..DLC (↑)	MAC_CO_DATA_IND	✓	✓
		MAC_CO_DTR_IND	✓	✓
		MAC_CON_CFM	-	✓
		MAC_CON_IND	✓	-
		MAC_DIS_IND	✓	✓
MASAP_PT MASAP_FT	L..MASAP..DLC..MAC (↓)	MAC_PAGE_REQ	✓	-
	L..MASAP..MAC..DLC (↑)	MAC_PAGE_IND	-	✓

[17] Una descripción detallada de cada señal puede obtenerse en la Especificación del Nivel MAC [ETS 300 175-3].

SPICUM servicio de publicaciones

Tabla 8.17: Listas de señales de las interfaces internas para los Sistemas de Pruebas del Nivel DLC entre el bloque `MAC_CCF` y el bloque `LINSER`.

Canal	Lista de Señales	Señal[18]	MProt_DLC	
			PT	FT
CCF_LINSER_FT CCF_LINSER_PT	L..CCF..LINSER (↓)	CSF_CO_DATA_REQ	✓	✓
		CSF_DBC_REL_REQ	✓	✓
		CSF_DBC_REQ	✓	✓
		CSF_INIT_REQ	✓	✓
		CSF_LINSER_RESET	✓	✓
		CSF_PAGE_INFO_FULL_REQ	✓	-
		CSF_PAGE_INFO_SHORT_REQ	✓	-
		CSF_SCAN_REQ	-	✓
		CSF_SET_HO_REQ	✓	✓
		CSF_TBC_BUF_REQ	✓	-
		CSF_TBC_REL_REQ	✓	✓
		CSF_TBC_REQ	-	✓
		CSF_TBC_RES_REQ	✓	-
	L..LINSER..CCF (↑)	CSF_CO_DATA_IND	✓	✓
		CSF_CO_DTR_IND	✓	✓
		CSF_DIS_IND	✓	✓
		CSF_FP_CAP_IND	-	✓
		CSF_HO_IND	✓	✓
		CSF_LINSER_ERROR	✓	✓
		CSF_LINSER_START	✓	✓
		CSF_NT_IND	-	✓
		CSF_PAGE_CFM	✓	-
		CSF_PAGE_FULL_IND	-	✓
		CSF_PAGE_SHORT_IND	-	✓
		CSF_RSSI_TABLE	✓	✓
		CSF_STAT_INFO_IND	-	✓
		CSF_TBC_CFM	✓	✓
		CSF_TBC_IND	✓	-
		CSF_TIMEOUT_IND	✓	✓

[18] Una descripción detallada de cada señal puede obtenerse en [SANC00a] y [SIT94B].

SPICUM servicio de publicaciones

Tabla 8.18: Listas de señales de las interfaces internas para los Sistemas de Pruebas del Nivel DLC entre el bloque MAC_CCF y la entidad de gestión LLME_MAC.

Canal	Lista de Señales	Señal[19]	MProt_DLC	
			PT	FT
MESAP_MAC_FT MESAP_MAC_PT	L..LLME..MAC_FT L..LLME..MAC_PT (→)	MAC_ME_CHAN_INFO_RES	✓	-
		MAC_ME_CON_ALL_REQ	✓	✓
		MAC_ME_DIS_REQ	✓	-
		MAC_ME_INFO_RES	✓	✓
		MAC_ME_MAD_HO_IND	-	✓
		MAC_ME_RESET_FT	✓	-
		MAC_ME_RESET_PT	-	✓
		MAC_ME_RFP_PRELOAD_REQ	✓	-
		MAC_ME_SINC_ERR_RES	-	✓
		MAC_ME_TX_BLIND_SLOT_REQ	✓	-
	L..MAC..LLME_FT L..MAC..LLME_PT (←)	MAC_ME_B_HO_ERR_IND	-	✓
		MAC_ME_CON_IND	✓	-
		MAC_ME_CON_REQ	-	✓
		MAC_ME_CHAN_INFO_IND	-	✓
		MAC_ME_CHAN_INFO_REQ	✓	-
		MAC_ME_DIS_IND	✓	✓
		MAC_ME_INFO_IND	✓	✓
		MAC_ME_SINC_ERR_IND	✓	✓
		MAC_ME_SINC_IND	-	✓

Tabla 8.19: Listas de señales de las interfaces internas para el Sistema de Pruebas del Nivel NWK de la PT.

Canal	Lista de Señales	Señal[20]
MCSAP_FT	L..MCSAP..DLC..MAC (↓)	Igual que en MProt_DLC_PT
	L..MCSAP..MAC..DLC (↑)	
MASAP_FT	L..MASAP..DLC..MAC (↓)	
CCF_LINSER_FT	L..CCF..LINSER (↓)	
	L..LINSER..CCF (↑)	
MESAP_MAC_FT	L..LLME..MAC_FT (→)	
	L..MAC..LLME_FT (←)	
MESAP_DLC_FT	L..LLME..DLC_FT (→)	LAPC_CON_IND
		LAPC_DIS_IND
		Lc_MAC_DIS_REQ
		Lc_MOD
	L..DLC..LLME_FT (←)	LAPC_DIS_REQ
		Lc_MAC_CON_IND
		Lc_MAC_DIS_IND

Sistema de Pruebas para el Nivel NWK

Las interfaces internas (Tabla 8.19) del Módulo de Protocolos incluyen cinco canales: MCSAP_FT, MASAP_FT, CCF_LINSER_FT, MESAP_MAC_FT y MESAP_DLC_FT. Los cuatro primeros canales transportan las mismas señales que en el caso de los Sistemas de Pruebas para el Nivel DLC, y desempeñan la misma función, con la salvedad de que los canales MCSAP_FT y MASAP_FT conectan ahora el bloque MAC_CCF con el bloque DLC. El quinto canal es nuevo y comunica el bloque DLC con el bloque de gestión LLME_FT

[19] Una descripción detallada de cada señal puede obtenerse en [SANC00a].

[20] Una descripción detallada de cada señal puede obtenerse en [TAPI00].

SPICUM servicio de publicaciones

Las señales de los canales `MESAP_MAC_FT` y `MESAP_DLC_FT` (Tabla 8.18) se han definido a partir de [ETS 300 175-4] y [ETS 300 175-3], respectivamente, donde aparece la lista de señales de gestión y la funcionalidad esperada de cada una, pero sin especificar sus parámetros ni su tipo. La definición de las primitivas de gestión del Nivel DLC se ha incluido en el archivo `Otras_primitivas_DLC.asp` y las del Nivel MAC en `Otras_primitivas_MAC.asp`.

Tabla 8.20: Primitivas de la interfaz con el Módulo de Capa Física.

Sentido	Señal	Código	Parámetros	MProt_xx	
				PT	FT
(↑)	CSF_CO_DATA_IND	0x07	intervalo, datos	✓	✓
	CSF_CO_DTR_IND	0x08	intervalo	✓	✓
	CSF_DIS_IND	0x0B	intervalo	✓	✓
	CSF_FP_CAP_IND	0x03	intervalo, macinfo, highinfo	-	✓
	CSF_HO_IND	0x0E	intervalo, hoinfo	✓	✓
	CSF_NT_IND	0x01	intervalo, rfpi	-	✓
	CSF_PAGE_CFM	0x0D	---	✓	-
	CSF_PAGE_FULL_IND	0x04	intervalo, datos_aviso_full	-	✓
	CSF_PAGE_SHORT_IND	0x05	intervalo, tipo, intervalo, portadora, datos_aviso	-	✓
	CSF_RSSI_TABLE	0x0F	portadora, nivel	✓	✓
	CSF_STAT_INFO_IND	0x02	intervalo, intervalo, statinfo, portadora, pscn	-	✓
	CSF_TBC_CFM	0x0A	intervalo	✓	✓
	CSF_TBC_IND	0x09	intervalo, tipo, fmidpmid	✓	-
	CSF_TIMEOUT_IND	0x20	intervalo	✓	✓
(↓)	CSF_CO_DATA_REQ	0x0A	intervalo, datos	✓	✓
	CSF_DBC_REL_REQ	0x02	intervalo	✓	✓
	CSF_DBC_REQ[21]	0x01	intervalo, portadora	-	✓
			intervalo, portadora, pscn	✓	✓
	CSF_INIT_REQ[19]	0x00	rfpi, macinfo, highinfo	✓	-
			---	-	✓
	CSF_PAGE_INFO_FULL_REQ	0x08	datos_aviso_full	✓	-
	CSF_PAGE_INFO_SHORT_REQ	0x07	tipo, intervalo, portadora, datos_aviso	✓	-
	CSF_SCAN_REQ	0x09	intervalo	-	✓
	CSF_SET_HO_REQ	0x0B	ahandover, bhandover, rssilow, rssilevel, txbuff	✓	✓
	CSF_TBC_BUF_REQ	0x06	intervalo, rxbuff, txbuff	✓	-
	CSF_TBC_REL_REQ	0x05	intervalo	✓	✓
	CSF_TBC_REQ	0x04	intervalo, tipo, fmidpmid	-	✓
	CSF_TBC_RES_REQ	0x03	intervalo, rxbuff, txbuff, fmidpmid	✓	-

8.5.1.2.3 Interfaz con el Módulo de Capa Física

Esta interfaz viene definida en la documentación de los módulos DECT [SIT94c]. Las primitivas que componen la interfaz, así como una descripción de los parámetros que

[21] Señales comunes a las Terminaciones Fija y Portátil pero con distintos parámetros

SPICUM servicio de publicaciones

transportan se muestran en la Tabla 8.20 y la Tabla 8.21. Las primitivas están agrupadas en sentido ascendente (↑), del Módulo de Capa Física al Módulo de Protocolos, y sentido descendente (↓), del Módulo de Protocolos al Módulo de Capa Física.

Tabla 8.21: Parámetros, y su descripción, de las primitivas de la interfaz con el Módulo de Capa Física.

Nombre	Tipo	Descripción
ahandover	`Integer`	Número máximo de fallos tolerados en el campo A de una trama de Nivel Físico antes de solicitar traspaso de portadora ([0..255]).
bhandover	`Integer`	Número máximo de fallos tolerados en el campo B de una trama de Nivel Físico antes de solicitar traspaso de portadora ([0..255]).
datos	`Octet_String`	SDU (*Service Data Unit*) de 5 octetos con los datos transmitidos/recibidos.
datos_aviso	`Octet_String`	Información de aviso para el Nivel DLC (SDU de 4 octetos, pero sólo los 20 bits menos significativos son útiles).
datos_aviso_full	`Octet_String`	Información de aviso para el Nivel DLC (SDU de 5 octetos).
fmidpmid	`Octet_String`	Identificador asociado a la conexión (5 octetos).
highinfo	`Octet_String`	Información de las capacidades del sistema de las capas superiores (SDU de 2 octetos).
hoinfo	`Integer`	Motivo que origina la petición de traspaso.
intervalo	`Integer`	Número de intervalo del canal físico ([0..11]).
macinfo	`Octet_String`	Información de las capacidades del sistema de Nivel MAC (SDU de 3 octetos).
mute	`Integer`	Número de fallos que debe producirse en el campo B antes de desactivarlo ([0..255]).
nivel	`Integer`	Nivel de señal ([0..63]).
portadora	`Integer`	Número de portadora en la que se buscará ([0..9]).
pscn	`Integer`	Número de la siguiente portadora de recepción libre ([0..9]).
rfpi	`Octet_String`	Información de identidad del sistema (SDU de 5 octetos).
rssilevel	`Integer`	Nivel de señal umbral para solicitar traspaso de portadora ([0..63]).
rssilow	`Integer`	Número de veces consecutivas que el nivel de señal debe estar por debajo del umbral para solicitar un traspaso ([0..255]).
rxbuff	`Integer`	Configuración de la zona de recepción de datos del Plano de Usuario.
statinfo	`Integer`	Información estática del sistema (SDU de 2 octetos).
tipo	`Integer`	Tipo del mensaje.
txbuff	`Integer`	Configuración de la zona de transmisión de datos del Plano de Usuario.

8.5.1.2.4 Interfaz adicional de Control

Las interfaces adicionales de control se han realizado a través del bloque de gestión, `LLME`. Las señales de estas interfaces se dividen en dos categorías:

- Señales utilizadas por las funciones TSO de los Juegos de Pruebas.
- Señales utilizadas por el Subsistema de Operación y Administración para el control del Subsistema Inferior.

Sistemas de Pruebas para el Nivel DLC

La funcionalidad requerida por las funciones TSO exige señales adicionales a las utilizadas en los Juegos de Pruebas. Se ha creado un nuevo canal, denominado

SPICUM servicio de publicaciones

TTCN_LLME_MAC_FT/PT[22], que incluye las señales mostradas en la Tabla 8.22. La semántica de estas señales es la siguiente:

- TTCN_MAC_BHO_PERM_REQ: Habilita/deshabilita el traspaso de portadora.
- TTCN_MAC_CHO_PERM_REQ: Habilita/deshabilita el traspaso de conexión.
- TTCN_MAC_FPCAP_CFM/IND/REQ/RES: Modifica el campo "Capacidades de la capa superior" que envía la Terminación Fija en los mensajes de difusión MAC.
- TTCN_MAC_RFPI_REQ: Indica el nuevo valor de RFPI (*Radio Fixed Part Identity*) a usar por los módulos radio.
- TTCN_MAC_TX_BLIND_SLOT_REQ: Solicita que se transmita de forma instantánea información de "*blind slot*".

Además, se han declarado otras señales para controlar el comportamiento del Subsistema Inferior desde el Subsistema de Operación y Administración (Tabla 8.23). La semántica de estas señales es la siguiente:

- CONFIG_FT: Configura los valores de la identidad, derechos de acceso y capacidades de la Terminación Fija. Provoca la reinicialización del sistema.
- RESET_FT/PT: Reinicializa el sistema.

Tabla 8.22: Lista de señales utilizadas por las funciones TSO.

Canal	Lista de Señales	Señal	MProt_DLC	
			PT	FT
TTCN_LLME_MAC_FT TTCN_LLME_MAC_PT	L..TTCN..LLME_MAC_FT L..TTCN..LLME_MAC_PT (↓)	TTCN_MAC_BHO_PERM_REQ	✓	-
		TTCN_MAC_CHO_PERM_REQ	✓	-
		TTCN_MAC_FPCAP_IND	✓	✓
		TTCN_MAC_FPCAP_REQ	✓	✓
		TTCN_MAC_RFPI_REQ	✓	-
		TTCN_MAC_TX_BLIND_SLOT_REQ	✓	-
	L..LLME_MAC..TTCN_FT L..LLME_MAC..TTCN_PT (↑)	TTCN_MAC_FPCAP_CFM	✓	✓
		TTCN_MAC_FPCAP_RES	✓	✓
		TTCN_MAC_NWK_IND	✓	-

Tabla 8.23: Lista de señales utilizadas desde el Subsistema de Operación y Administración.

Canal	Lista de Señales	Señal	MProt_DLC	
			PT	FT
ENV_LLME_FT ENV_LLME_PT	L..ENV..LLME_FT L..ENV..LLME_PT (↓)	CONFIG_FT	✓	-
		RESET_FT	✓	-
		RESET_PT	-	✓

Sistema de Pruebas para el Nivel NWK

Las señales utilizadas por las funciones TSO incluyen las señales utilizadas en el Sistema de Pruebas para el Nivel DLC (Tabla 8.22), y las señales adicionales

[22] Este canal es el canal INTERNO indicado en el Capítulo 4, Sección 4.5.1.2.4.

SPICUM servicio de publicaciones

LLME_DLC_LINKPRESENT_REQ/CFM[22] (Tabla 8.24). Estas últimas permiten comprobar si hay un enlace activo con la IUT. Las señales utilizadas por el Subsistema de Operación y Administración son las mismas que en los Sistemas de Pruebas para el Nivel DLC (Tabla 8.23).

Tabla 8.24: Señales utilizadas por las funciones TSO añadidas por el Sistema de Pruebas del Nivel NWK.

Canal	Lista de Señales	Señal
TTCN_LLME_DLC_FT	L..TTCN..LLME_DLC_FT (↓)	LLME_DLC_LINKPRESENT_REQ
	L..LLME_DLC..TTCN_FT (↑)	LLME_DLC_LINKPRESENT_CFM

8.5.1.3 Plan de Pruebas

Este Plan de Pruebas recoge las comprobaciones que se van a realizar sobre los elementos realizados en cada etapa. Se han incluido desde pruebas sobre elementos básicos como procedimientos o procesos, hasta pruebas que contrasten los Sistemas de Pruebas resultantes.

8.5.1.3.1 Pruebas Unitarias

Las pruebas unitarias se realizarán con el simulador de SDL. Son pruebas que no es posible especificar en este punto, ya que aún no se ha realizado el diseño detallado de cada bloque, pero incluirán la verificación de los elementos que los compongan:

- Procedimientos.
- Transiciones entre estados.
- Procesos.

8.5.1.3.2 Pruebas de Nivel

Los Niveles que se necesitan para implementar los Módulos de Protocolos son el Nivel MAC[23] y el Nivel DLC. Las pruebas definidas sobre cada nivel se describen a continuación.

Nivel MAC

Las pruebas del Nivel MAC (Tabla 8.25) son aplicables a ambas Terminaciones y cubren aspectos de funcionamiento normal, respuesta a fallos y transferencia correcta de datos.

[23] La parte superior del Nivel MAC, que incluye la funcionalidad CCF (*Cluster Control Functions*).

SPICUM servicio de publicaciones

Tabla 8.25: Lista de Pruebas de Nivel para el Nivel MAC (ambas Terminaciones).

Prueba	Objetivo
M1	Establecimiento de una conexión.
M2	Envío de datos en una conexión.
M3	Liberación de una conexión.
M4	Llegada de una petición de conexión durante un traspaso.
M5	Reinicialización con conexiones de tráfico abiertas.
M6	Solicitud de traspaso de conexión sin haber canales de tráfico disponibles.
M7	Solicitudes consecutivas de traspaso de conexión.
M8	Intento de conexión con identificación (FMID[24]) errónea.
M9	Envío de información de capacidades físicas no GAP.

Nivel DLC

La comprobación del modelo del Nivel DLC hará uso de dos categorías de pruebas:

a) En primera instancia, se ha someterá el Nivel a una batería de pruebas que examinan algunos aspectos del comportamiento básico del mismo; con ello se obtiene confianza en el comportamiento habitual del modelo. Estas pruebas se implementan en el simulador. La Tabla 8.26 muestra las pruebas a realizar sobre la implementación de la Terminación Fija y la Tabla 8.27 hace lo propio para la Terminación Portátil.

b) A continuación, se ejecutarán sobre el modelo las Pruebas de Conformidad de Protocolo publicadas para el Nivel; para ello se utilizarán los Juegos de Pruebas Abstractas que se han obtenido en el apartado 8.4.1.

Tabla 8.26: Lista de Pruebas básicas de Nivel para el Nivel DLC de la Terminación Fija.

Prueba	Objetivo
DF1	Establecimiento del enlace iniciado por FT.
DF2	Establecimiento del enlace iniciado por PT.
DF3	Envío de una PDU tipo `CC-SETUP` en el enlace.
DF4	Recepción de una trama I con `PDU_CC_ALERTING` en el enlace.
DF5	Traspaso durante la recepción de datos en el enlace.
DF6	Envío de `PDU_CC_SETUP` con error por vencimiento.
DF7	Liberación normal de la conexión iniciada por MAC.

[24] *Fixed MAC Identity*.

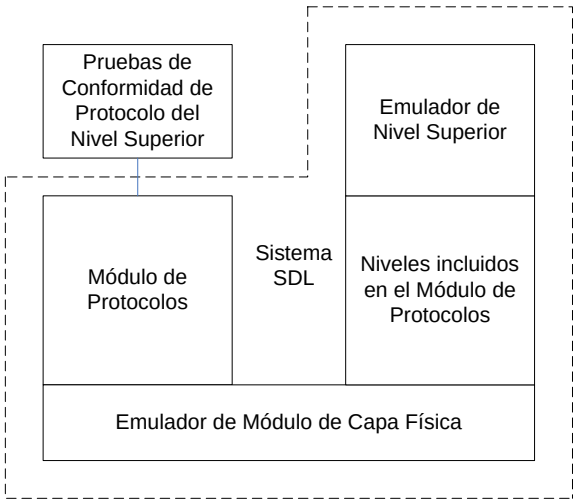
SPICUM servicio de publicaciones

Tabla 8.27: Lista de Pruebas básicas de Nivel para el Nivel DLC de la Terminación Portátil.

Prueba	Objetivo
DP1	Establecimiento de conexión como respuesta a la primitiva `PAGE-IND`.
DP2	Liberación normal de la conexión iniciada por NWK.
DP3	Envío de una `PDU_CC_ALERTING`.
DP4	Establecimiento de conexión iniciado por PT.
DP5	Traspaso durante el envío de datos.
DP6	Recepción de una trama I con `PDU_CC_SETUP` en PT.
DP7	Envío de `PDU_AUTH_REQUEST` con error por vencimiento.
DP8	Liberación normal mientras se envían datos.

8.5.1.3.3 Pruebas de Módulo

Estas pruebas se realizan para comprobar el Módulo de Protocolos de cada Sistema de Pruebas. La comprobación se realizará ejecutando las pruebas de conformidad del nivel a que corresponde el Sistema de Pruebas en que se incluye el Módulo de Protocolos sobre una emulación de la Implementación Bajo Prueba; los módulos aún no integrados, por ejemplo el Módulo de Capa Física, serán sustituidos por emuladores. Tanto el Módulo de Protocolos, como la Implementación Bajo Prueba y los emuladores forman parte del mismo sistema SDL, como muestra la Figura 8.10-a. La Figura 8.10-b indica los Juegos de Pruebas utilizados.



(a)

Módulo de Protocolos	Juego de Pruebas
`MProt_DLC_PT`	`DLC-PT`
`MProt_DLC_FT`	`DLC-FT`
`MProt_NWK_PT`	`NWK-PT`

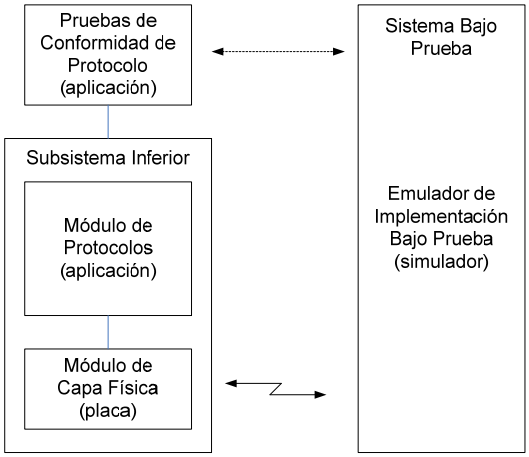
(b)

Figura 8.10: (a) Estructura genérica de las Pruebas de Módulo y (b) Juegos de Pruebas implicados.

8.5.1.3.4 Pruebas de Subsistema

Las Pruebas de Subsistema (Figura 8.11) son las mismas que las Pruebas de Módulo, pero empleando la implementación del Módulo de Capa Física. Esto implica que tanto el Sistema de Pruebas como el emulador de la IUT deben funcionar en tiempo real. Se utilizarán los Juegos de Pruebas correspondientes a cada Subsistema Inferior.

SPICUM servicio de publicaciones



(a)

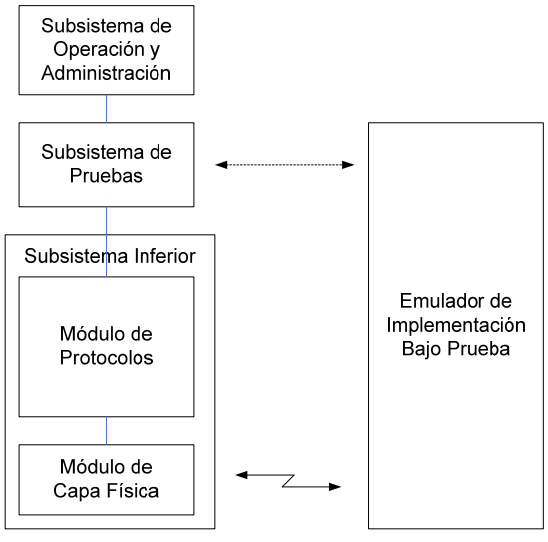
Sistema de Pruebas	Juego de Pruebas	Emulador de IUT
SP_DLC_PT	DLC-PT	PT
SP_DLC_FT	DLC-FT	FT
SP_NWK_PT	NWK-PT	PT

(b)

Figura 8.11: (a) Estructura genérica de las Pruebas de Subsistema y (b) Juegos de Pruebas y emuladores de IUT implicados.

8.5.1.3.5 Pruebas de Sistema

Las Pruebas de Sistema se ejecutan una vez realizada la integración de los distintos Subsistemas. Para verificar el correcto funcionamiento de los Sistemas de Pruebas se hará uso de un emulador de IUT, como se muestra en la Figura 8.12-a, que incluya el Nivel a que corresponde el Sistema de Pruebas. Las Pruebas a ejecutar corresponden a las de los correspondientes Juegos de Pruebas, como en el caso de las Pruebas de Subsistema; la ejecución correcta de las Pruebas permitirá asegurar un comportamiento adecuado.



(a)

Sistema de Pruebas	Subsistema de Pruebas	Emulador de IUT
SP_DLC_PT	Ets_dlc_pt.exe	PT
SP_DLC_FT	Ets_dlc_ft.exe	FT
SP_NWK_PT	Ets_nwk_pt.exe	PT

(b)

Figura 8.12: (a) Estructura genérica de las Pruebas de Sistema y (b) Juegos de Pruebas y emuladores de IUT implicados.

SPICUM servicio de publicaciones

8.5.2 Diseño del Módulo de Protocolos

En primer lugar, se describen los distintos bloques que constituyen cada Módulo de Protocolos, indicando los procesos que incluyen y las interfaces definidas (Sección 8.5.2.1). A continuación, se explica el comportamiento de cada uno de estos procesos, mostrando su diagrama de estados y las señales que pueden recibir (Sección 8.5.2.2). Finalmente, se comenta cómo se han realizado las Pruebas de Nivel, incluyendo los sistemas que se han construido para la realización de dichas pruebas (Sección 8.5.2.3).

Los apartados del Diseño de la Estructura (Sección 8.5.2.1) y del Diseño Detallado (Sección 8.5.2.2) se han organizado separando entre los elementos correspondientes a los Sistemas de Pruebas para la Terminación Portátil y el Sistema de Pruebas para la Terminación Fija.

8.5.2.1 Diseño de la Estructura

El diseño de la estructura interna de cada bloque ha tratado de seguir la estructura propuesta por la norma que describe su comportamiento. Esto no ha sido posible en el Nivel de Gestión, pues la descripción de su comportamiento se encuentra distribuida en las normas correspondientes a cada uno de los niveles y sólo proporciona información sobre los servicios que debe ofrecer. Como criterios de diseño, sólo se han definido procesos que se corresponden con un grupo funcional claramente indicado en la norma y se ha evitado el uso de estados dentro de los procedimientos.

Esta Sección se ha organizado de forma que en primer lugar se describen los bloques correspondientes a los Sistemas de Pruebas para la Terminación Portátil (`SP_DLC_PT` y `SP_NWK_PT`), bloques en el rol de Terminación Fija, y, seguidamente, los bloques del Sistema de Pruebas para la Terminación Fija (`SP_DLC_FT`), bloques actuando como Terminación Portátil. Como la estructura es muy similar para ambas Terminaciones, en los bloques que actúan como Terminación Portátil sólo se describen las diferencias respecto a los que actúan como Terminación Fija.

Los siguientes apartados presentan la estructura de cada uno de los bloques, con indicación expresa de los procesos que contienen y una descripción de la funcionalidad que cada uno debe implementar. La lista completa de canales, procesos, rutas de señales y listas de señales definidos en esta fase para cada bloque se adjunta en el Anexo G[25].

8.5.2.1.1 Sistemas de Pruebas para la Terminación Portátil

A continuación se describe la estructura de los bloques utilizados en los Sistemas de Pruebas para la Terminación Portátil. La Figura 8.13 muestra cuáles de estos bloques forman parte de cada Sistema de Pruebas.

[25] El objetivo es presentar en este apartado sólo una descripción básica, no exhaustiva, con objeto de presentar los componentes utilizados en las siguientes actividades.

SPICUM servicio de publicaciones

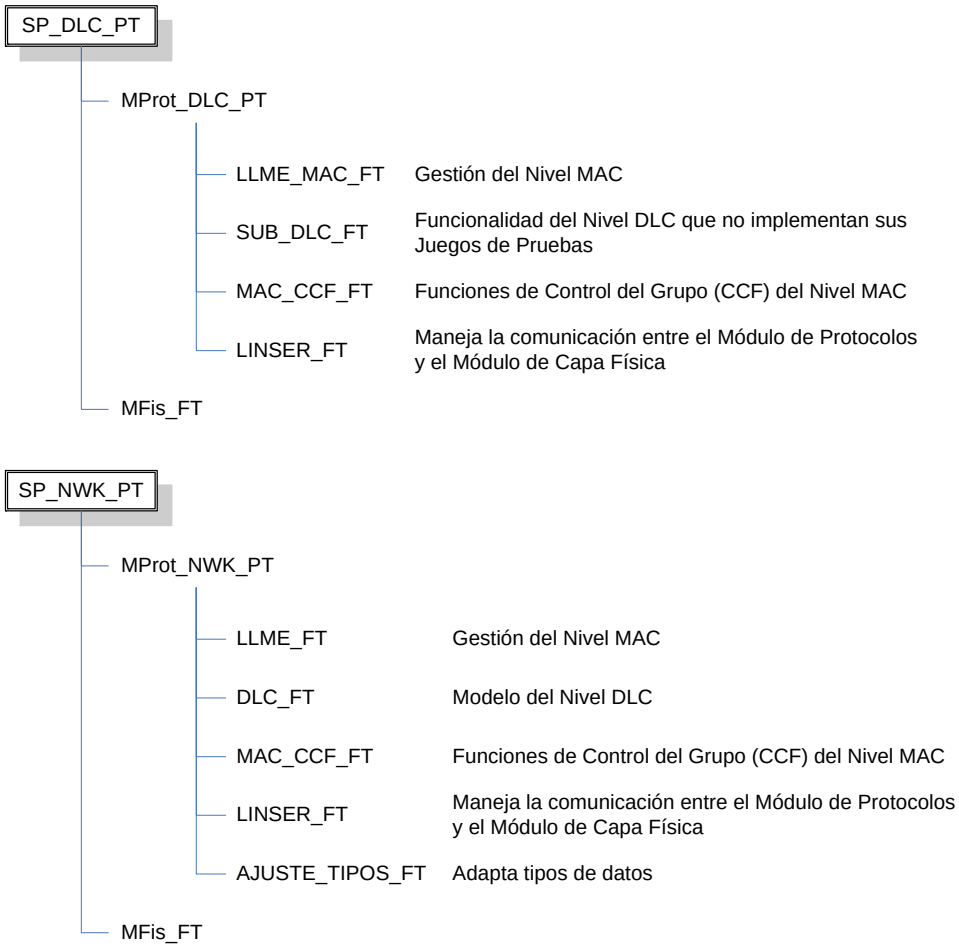


Figura 8.13: Bloques pertenecientes al Módulo de Protocolos de cada Sistema de Pruebas para la Terminación Portátil.

Bloque MAC_CCF_FT

El bloque `MAC_CCF_FT` (Figura 8.14) realiza las Funciones de Control del Grupo (CCF) del Nivel MAC. Su funcionalidad está desglosada en Servicio de Difusión y Servicio de Multiportadora.

El Servicio de Difusión (BMC - *Broadcast Message Control*), descendente en la Terminación Fija, se ha realizado mediante el proceso `BMC`. Este proceso gestiona los mensajes de difusión y las portadoras necesarias para su transmisión, además de realizar tareas de control de errores y de sincronización con el Nivel Físico. La comunicación con el Nivel DLC se realiza a través del canal `MASAP_FT`. También se encarga de la configuración del bloque a través del canal `MESAP_MAC_FT`.

El Servicio de Multiportadora (MBC – *Multi-Bearer Control*) se ha implementado mediante tres procesos: el Servidor de Conexiones (`MBC_CTRL`), el Gestor de Conexiones de Tráfico (`MBC`) y el Selector de Conexiones (`MBC_SELEC`). La funcionalidad de cada uno es la siguiente:

- `MBC_CTRL`: Es responsable de la creación y liberación de las conexiones de tráfico, creando una instancia del proceso `MBC` asociada a cada petición de conexión recibida; el establecimiento se acepta si el Nivel de Gestión lo autoriza. También redirecciona los mensajes del Nivel DLC a la instancia

SPICUM servicio de publicaciones

oportuna, para lo que debe mantener una lista de las conexiones abiertas; la comunicación con el Nivel DLC se realiza a través del canal `MCSAP_FT`.

- `MBC`: Este proceso es creado dinámicamente y se encarga de crear y gestionar una conexión de datos. Pueden existir hasta dos instancias[26].
- `MBC_SELEC`: Actúa como multiplexor de las señales del nivel inferior, seleccionando la instancia adecuada del proceso `MBC` para su procesamiento.

La interacción del bloque con el Módulo de Capa Física se realiza a través del canal `CCF_LINSER_FT`. La gestión del bloque se realiza a través del canal `MESAP_MAC_FT`. Las señales se encaminan al proceso `BMC` si están relacionadas con información de difusión, por ejemplo la información a transmitir de identidad de la Terminación Fija, y al proceso `MBC` si están relacionadas con el Servicio de Multiportadora, por ejemplo información de una nueva conexión y solicitudes de autorización para aceptarla.

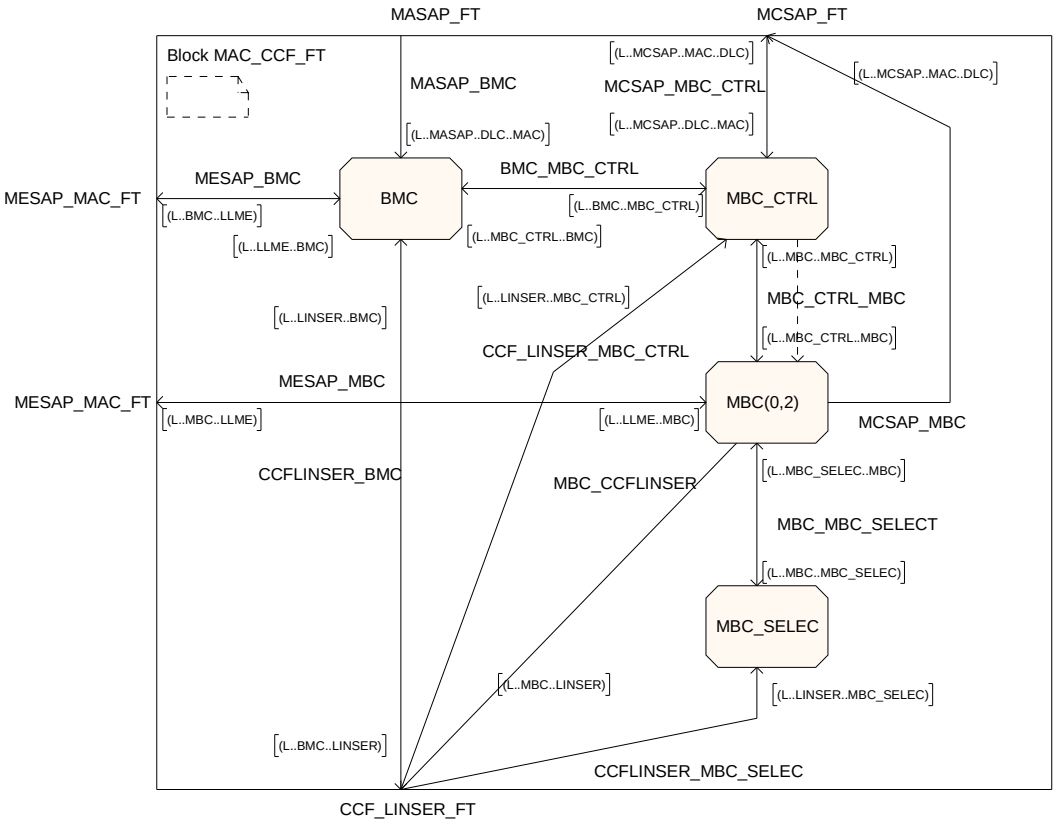


Figura 8.14: Modelo del Bloque `MAC_CCF_FT`.

Las señales de las listas `L..LINSER..CCF` y `L..CCF..LINSER` se dividen según el tipo de servicio. Las relacionadas con el Servicio de Difusión circulan por la ruta `CCFLINSER_BMC`. Las señales relacionadas con el Servicio de Multiportadora utilizan diferentes rutas según el sentido y su función. En recepción van hacia el proceso `MBC_CTRL` si son solicitudes de nuevas conexiones[27] y al proceso `MBC_SELEC` si son de

[26] Esta limitación es debida a las limitaciones de las placas.

[27] Al no haber una conexión establecida, no existirá ninguna instancia de proceso `MBC` y el proceso `MBC_SELEC` no estará configurado para poder encaminar la señal.

SPICUM servicio de publicaciones

conexiones ya existentes, para que éste las encamine al proceso `MBC` apropiado. En transmisión son sólo enviadas por las instancias del proceso `MBC`.

Bloque DLC_FT

Este bloque implementa el Nivel DLC del Sistema de Pruebas. Se ha dividido en dos bloques (Figura 8.15): el bloque `BROADCAST_FT`, que procesa la información de difusión, y el bloque `LINK_SERVICE_FT`, que se encarga del servicio de enlace de datos. Ambos bloques se comunican tanto con el nivel adyacente superior como con el inferior; además, el bloque `LINK_SERVICE_FT` se comunica con el bloque de gestión, a través del canal `ME_SAP`. Los procedimientos comunes a ambas Terminaciones están implementados en el paquete `Comun_DLC` (Apéndice I, Sección I.3.1).

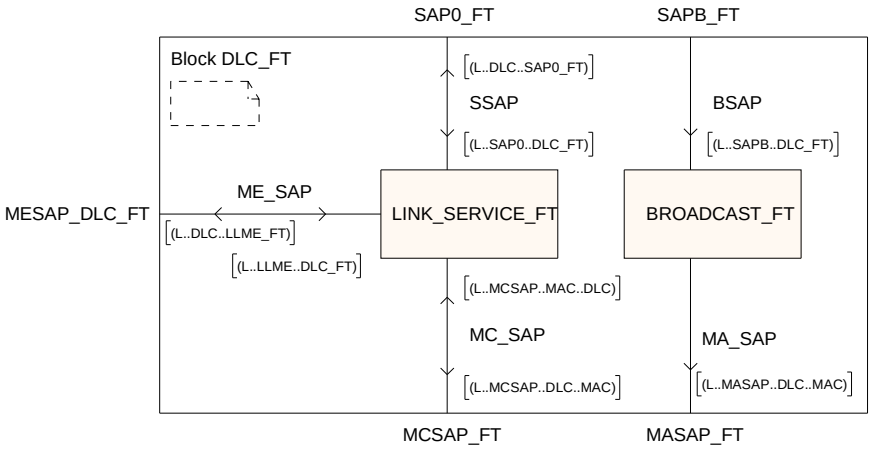


Figura 8.15: Modelo del Bloque `DLC_FT`.

El bloque `LINK_SERVICE_FT` permite el establecimiento simultáneo de más de un enlace. El par de instancias `LAPC_FT` – `Lc_FT` encargado de atender a cada enlace se crea dinámicamente cuando llega la solicitud de un nuevo enlace; el proceso `CTRL_FT` se encarga de crear estas instancias. La división de funcionalidad entre los cuatro procesos que lo forman (Figura 8.16-a) ha seguido los principios empleados en la norma [ETS 300 175-4]:

- `CTRL_FT`: gestiona la conexión y desconexión de los canales MAC.
- `LAPC_FT`: es el proceso responsable de las funciones de control del enlace.
- `Lc_FT`: proceso encargado de la fragmentación de tramas en unidades MAC y de la verificación de la suma de comprobación.
- `SignalROUTER`: encamina las primitivas desde el Nivel de Red hacia la instancia apropiada (`LAPC_FT` o `Lc_FT`).

El bloque `BROADCAST_FT` (Figura 8.16-b) es más simple, y sólo contiene un proceso (`Lb_FT`). Su funcionalidad se limita a convertir entre una primitiva del Nivel de Red y la correspondiente del Nivel de Acceso al Medio.

El bloque `DLC_FT` también codifica y descodifica las PDUs del Nivel de Red; el paquete `Comun_DLC` contiene un procedimiento de codificación y otro de descodificación para cada una de las posibles PDUs. Para codificar una PDU, se invoca el procedimiento

SPICUM servicio de publicaciones

Conv_Primi_PDU_NWK, que identifica el tipo de PDU y procede a codificarla utilizando el procedimiento apropiado (ej. Conv_Primi_PDU_CC_CONNECT). De igual forma, para descodificar se invoca el procedimiento Conv_Octet_PDU_NWK, que, tras determinar el tipo de PDU recibida, invoca el procedimiento de descodificación adecuado (ej. Conv_Octet_PDU_CC_CONNECT_ACK). La implementación de estas funciones está descrita en el Capítulo 5, Sección 5.5.2.

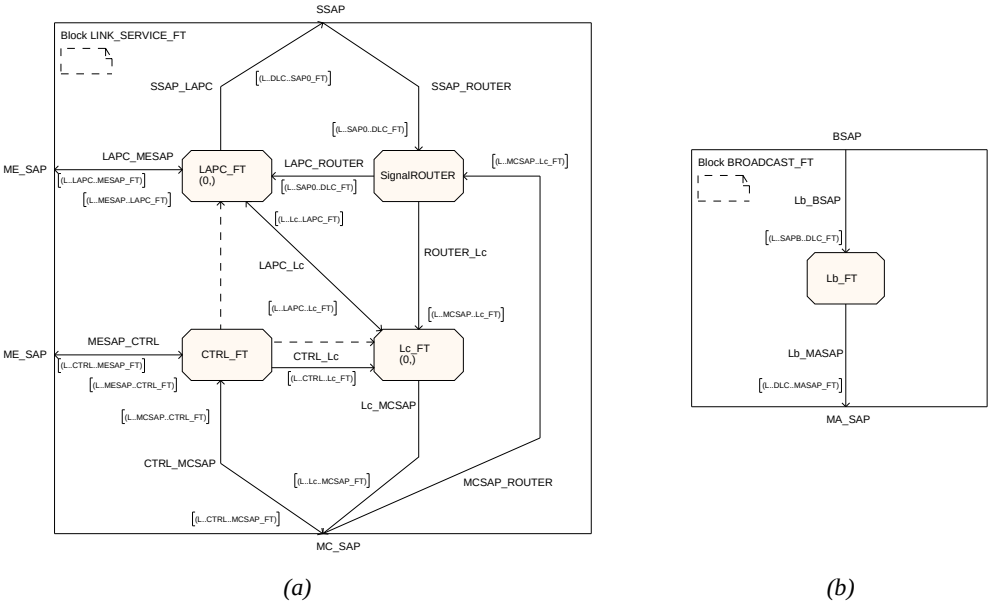


Figura 8.16: Modelo del (a) Bloque LINK_SERVICE_FT y (b) Bloque BROADCAST_FT.

Bloque SUB_DLC_FT

El bloque SUB_DLC_FT es el encargado de realizar la funcionalidad del Nivel DLC que no implementan sus Juegos de Pruebas. Realiza la fragmentación y recombinación de las tramas DLC y codifica (y descodifica) la información que transportan algunas de las señales. Este bloque se comunica con el Subsistema de Pruebas a través del canal LMAC, y con el Nivel MAC a través de los canales MASAP_FT y MCSAP_FT.

Las señales que deben ser codificadas son las señales de datos. Las señales de datos de las Pruebas (MAC_DATA_REQ, MAC_DATA_IND), son distintas de las señales de datos del Nivel MAC indicadas en la norma (MAC_CO_DATA_REQ, MAC_CO_DATA_IND, MAC_CO_DTR_IND). Las primeras llevan un campo de tipo PDU, que puede contener cualquier PDU utilizada en las Pruebas; sin embargo, las segundas sólo incluyen un campo de tipo Octet_String, de tamaño cinco octetos. Por tanto, hace falta una conversión entre ambas representaciones, la cual se realiza en este bloque. Existe un procedimiento para codificar y descodificar cada una de las posibles PDUs que pueden transportar las primitivas MAC_DATA_REQ y MAC_DATA_IND, respectivamente.

Este bloque está constituido por tres procesos:

- ConversorTTCN: Realiza la conversión entre las primitivas de datos usadas por el Subsistema de Pruebas y las primitivas del Nivel MAC.
- Cuasi_Lc: Realiza unas funciones similares al proceso Lc_FT del bloque DLC_FT, encargándose de la delimitación, fragmentación y recombinación de las

SPICUM servicio de publicaciones

tramas, la verificación de la suma de comprobación y el control de flujo (`MAC_CO-DTR_IND`).

- `Signal_RTX`: Copia las señales que no requieren procesamiento en la salida.

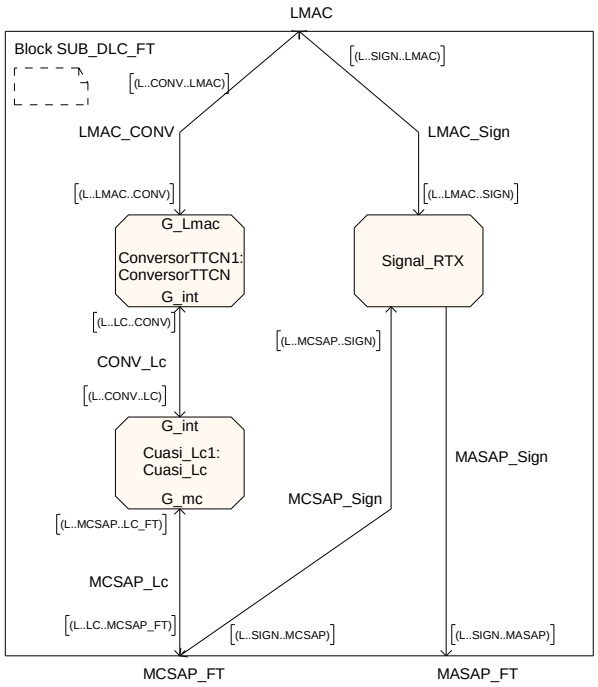


Figura 8.17: Modelo del Bloque `SUB_DLC_FT`.

Bloque LLME_MAC_FT

El bloque `LLME_MAC_FT` (Figura 8.18) se encarga de la gestión del Nivel MAC. Este bloque almacena la información de configuración, recibe los datos de las peticiones de nueva conexión, determina si está permitido su establecimiento, mantiene una lista de las conexiones activas y recibe las solicitudes de configuración durante la ejecución de las Pruebas. La comunicación con el Nivel MAC se realiza a través del canal `MESAP_LLME_MAC_FT`. La comunicación con el Subsistema de Pruebas se realiza a través del canal `TTCN_LLME_MAC_FT`, para configurar los parámetros, y el canal `ENV_LLME_MAC_FT`, que permite un cierto control sobre el comportamiento del sistema durante la ejecución, por ejemplo, para resetear el Subsistema Inferior.

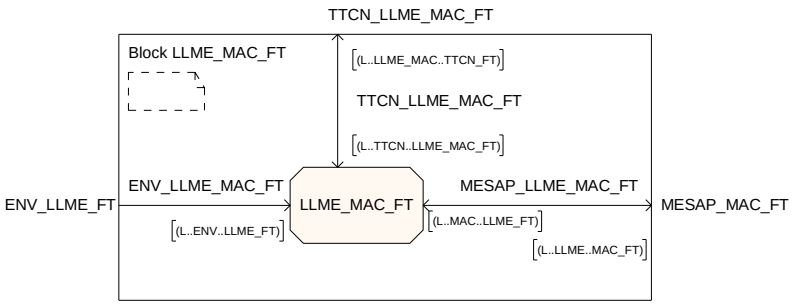


Figura 8.18: Modelo del Bloque `LLME_MAC_FT`.

SPICUM servicio de publicaciones

Bloque LLME_FT

El bloque `LLME_FT` (Figura 8.19) realiza el Nivel de Gestión en el Sistema de Pruebas del Nivel NWK. Se encuentra estructurado en dos procesos, cada uno de los cuales gestiona un nivel diferente: el proceso `LLME_DLC_FT` gestiona el Nivel DLC y el proceso `LLME_MAC_FT` gestiona el Nivel MAC. Ambos bloques son independientes, por lo que no existe ruta de comunicación entre ellos.

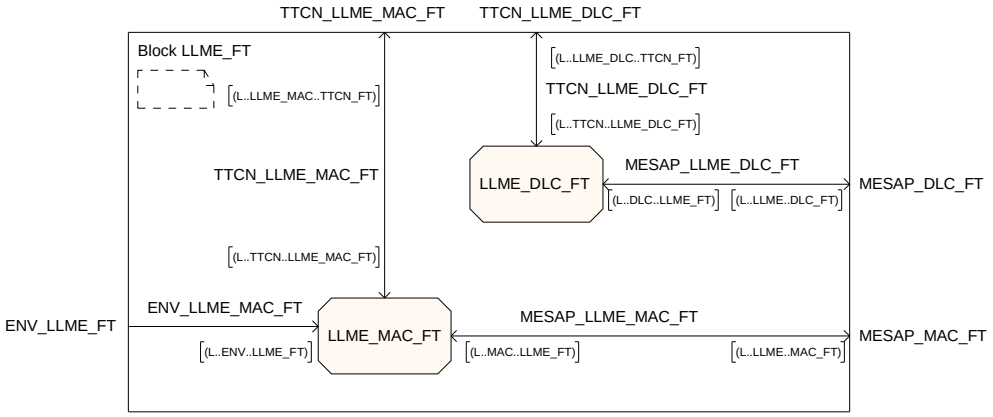


Figura 8.19: Modelo del Bloque `LLME_FT`.

El proceso `LLME_DLC_FT` es responsable de gestionar (establecer, liberar y modificar) las conexiones MAC, controlar el traspaso de las conexiones, almacenar y proporcionar identidades (parámetros PMID[28], MCEI[29], …) y mantener una lista de las conexiones en curso. Puede controlar múltiples enlaces y hasta dos conexiones MAC para cada enlace (para traspasos). Se comunica con el Subsistema de Pruebas a través del canal `TTCN_LLME_DLC_FT`, y con el Nivel DLC a través del canal `MESAP_LLME_DLC_FT`.

El proceso `LLME_MAC_FT` es el mismo que el del bloque `LLME_MAC_FT` (ver apartado anterior).

Bloque AJUSTE_TIPOS_FT

La inclusión del bloque `AJUSTE_TIPOS_FT`[30] (Figura 8.20) ha sido debida a la elección de las interfaces de los Módulos de Protocolos de los Sistemas de Pruebas para el Nivel de Red. En la definición de Pruebas del Nivel de Red para las Terminaciones Fija y Portátil aparecen tipos de igual nombre pero distinta definición. No se ha considerado conveniente introducir esta asimetría en la definición de tipos SDL, optando por unificar los tipos empleados en ambas Terminaciones. Sin embargo, aunque internamente los tipos son iguales, en la interfaz con el Subsistema de Pruebas hace falta utilizar los tipos apropiados. El bloque `Ajuste_Tipos_FT` se encarga de realizar esta conversión para los tipos de datos `PDU_IDENTITY_REPLY` y `PDU_AUTH_REJECT`.

[28] *Portable part MAC Identity*.

[29] *MAC Connection Endpoint Identification*.

[30] Este bloque no añade funcionalidad al Sistema y sólo actúa como conversor de algunos tipos de datos. Por eso, no se ha incluido en la Figura 8.9.

SPICUM servicio de publicaciones

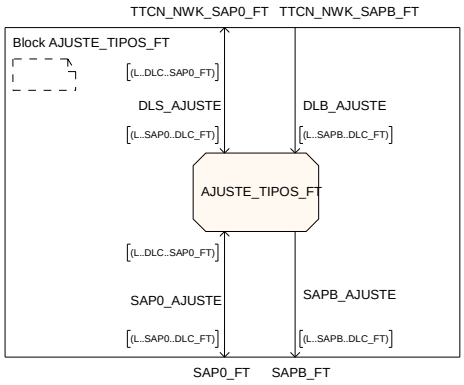


Figura 8.20: Modelo del Bloque `AJUSTE_TIPOS_FT`.

Bloque LINSER_FT

La misión del bloque `LINSER_FT` (Figura 8.21) es manejar la comunicación entre el Módulo de Protocolos y el Módulo de Capa Física y codificar/descodificar las señales transmitidas/recibidas. Contiene un único proceso, `LINSER_FT`, que se comunica con las funciones de nivel superior del MAC a través del canal `CCF_LINSER_FT`. La comunicación con el Módulo de Capa Física se realiza a través del manejador de los módulos (Sección 8.5.3.2.1). Este bloque hace uso de los procedimientos incluidos en el Paquete `Esc_Lec_Serie` (Apéndice I, Sección I.3.4).

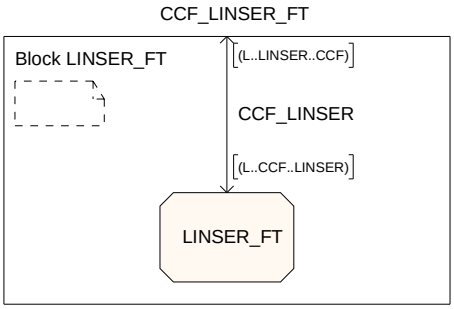


Figura 8.21: Modelo del Bloque `LINSER_FT`.

8.5.2.1.2 Sistemas de Pruebas para la Terminación Fija

Los siguientes apartados describen las diferencias de la estructura de los bloques utilizados en los Sistemas de Pruebas para la Terminación Fija, rol de Terminación Portátil, con respecto a la de los bloques empleados en los Sistemas de Pruebas para la Terminación Portátil, rol de Terminación Fija. No se han incluido diagramas SDL por ser muy similares a los ya presentados. La Figura 8.22 muestra cuáles de estos bloques forman parte de cada Sistema de Pruebas.

SPICUM servicio de publicaciones

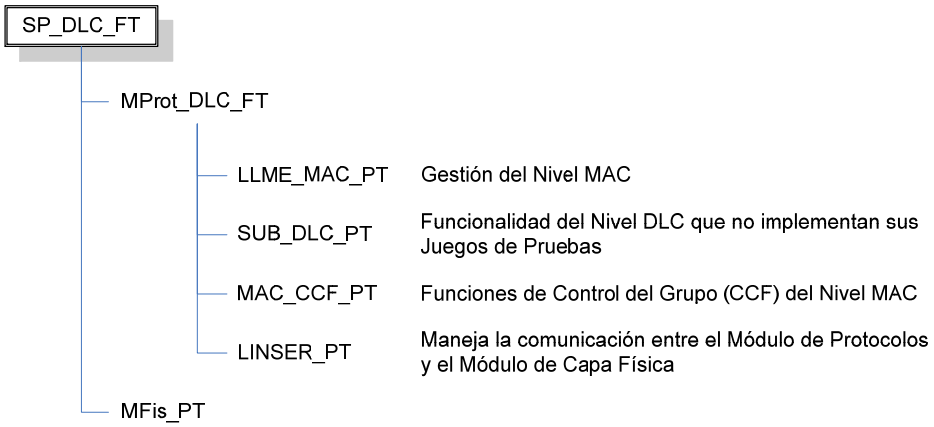


Figura 8.22: Bloques pertenecientes al Módulo de Protocolos del Sistema de Pruebas para la Terminación Fija.

Bloque MAC_CCF_PT

La estructura del bloque `MAC_CCF_PT` es similar a la del correspondiente bloque en la Terminación Fija, `MAC_CCF_FT` (Figura 8.14). La diferencia entre ambos estriba en el sentido de las rutas de señales del Servicio de Difusión, que son ascendentes en el caso de la Terminación Portátil. Además, según la especificación GAP, sólo puede haber una conexión activa en la Terminación Portátil y dos si se está produciendo un traspaso; por ello, existirá un máximo de dos instancias del proceso `MBC`.

En cuanto al encaminamiento de señales, la diferencia más significativa es que el Nivel de Gestión no se comunica con el proceso `MBC` sino con el proceso `MBC_CTRL`. Esto ocurre así porque, al iniciarse las conexiones desde la Terminación Portátil, se debe crear una instancia del proceso `MBC` que atienda dicha conexión; para ello, el proceso `MBC_CTRL` debe antes solicitar al Nivel de Gestión la autorización oportuna.

Bloque SUB_DLC_PT

La estructura del bloque `SUB_DLC_PT` es la misma que la del bloque `SUB_DLC_FT` (Figura 8.17), variando tan sólo el conjunto de señales que circula por el proceso `Signal_RTX` y el sentido de la ruta de señales correspondiente al Servicio de Difusión (canal `MASAP_PT`), descendente en este caso.

Bloque LLME_MAC_PT

La estructura del bloque `LLME_MAC_PT` es similar a la del bloque `LLME_MAC_FT` (Figura 8.18). En la Terminación Portátil se procesa la información de interés recibida a través del Servicio de Difusión.

Bloque LINSER_PT

La estructura del bloque `LINSER_PT` es idéntica a la del bloque `LINSER_FT` (Figura 8.21), aunque no hace uso de todas las señales como ya se indicó.

8.5.2.1.3 Paquetes auxiliares

Para organizar adecuadamente el diseño, se ha definido un conjunto de paquetes auxiliares que contienen elementos utilizados por los bloques presentados en los

SPICUM servicio de publicaciones

apartados anteriores[31]. Un primer grupo de paquetes está formado por aquellos paquetes comunes a todos los Módulos de Protocolos, para ambas Terminaciones. Otros paquetes se usan sólo en una Terminación. Por último, algún otro paquete es específico de un Sistema de Pruebas.

Los paquetes que se han definido son los siguientes:

- `Tipos_SDL`: Contiene la referencia a los archivos de tipos empleados en los distintos niveles de los Módulos de Protocolos. De forma general, se puede distinguir entre 3 tipos de ficheros: ficheros de tipos de primitivas (`.asp`), ficheros de tipos de PDUs (`.pdu`) y ficheros de tipos abstractos de datos (`.tad`).
- `Senales`: Contiene la declaración de todas las señales empleadas en los Sistemas de Pruebas, así como la mayoría de las listas de señales utilizadas en los diagramas (algunas no se incluyen por ser incompatibles, otras por ser internas, etc.). Tanto las señales como las listas de señal se encuentran clasificadas por Nivel (Módulo de Capa Física, MAC, DLC y NWK) y por funcionalidad (interfaz, intercambio entre entidades de la misma capa, etc.).
- `Senales_MAC_PT`: Contiene la declaración de las señales y listas de señales empleadas en el Nivel MAC de la Terminación Portátil.
- `Senales_MAC_FT`: Contiene la declaración de las señales y listas de señales empleadas en el nivel MAC específicas de la Terminación Fija.
- `Comun_DLC`: Incluye los procedimientos comunes a las Terminaciones FT y PT del Nivel DLC, así como los parámetros de configuración. Incluye procedimientos para la codificación y descodificación de PDUs del Nivel de Red.
- `Comun_MAC`: Este paquete incluye procedimientos comunes a las Terminaciones PT y FT utilizados para la gestión de las listas de conexiones y los parámetros de configuración del Nivel MAC.
- `Esc_Lec_Serie`: Ofrece procedimientos de conversión de datos del Módulo de Capa Física.

Tabla 8.28: Paquetes utilizados en el diseño de cada Módulo de Protocolos.

		Módulos de Protocolos		
		MProt_DLC_PT	**MProt_DLC_FT**	**MProt_NWK_PT**
Paquetes	**Tipos_SDL**	✓	✓	✓
	Senales	✓	✓	✓
	Senales_MAC_PT	-	✓	-
	Senales_MAC_FT	✓	-	✓
	Comun_MAC	✓	✓	✓
	Comun_DLC	-	-	✓
	Esc_Lec_Serie	✓	✓	✓
	Sub_DLC	✓	✓	-

[31] Para reducir el número de paquetes, algunos de ellos incluyen declaraciones de elementos no utilizadas en algún

SPICUM servicio de publicaciones

- `Sub_DLC`: Contiene los procesos de adaptación de primitivas entre el Subsistema de Pruebas y el Nivel MAC. Se emplea en los Módulos de Protocolos de los Sistemas de Pruebas del Nivel DLC.

La Tabla 8.28 indica qué paquetes forman parte de cada uno de los Módulos de Protocolos; la Figura 8.23 muestra la vista superior de la estructura del Módulo de Protocolos `MProt_DLC_FT`. El listado de los procedimientos de cada paquete se incluye en el Anexo G.

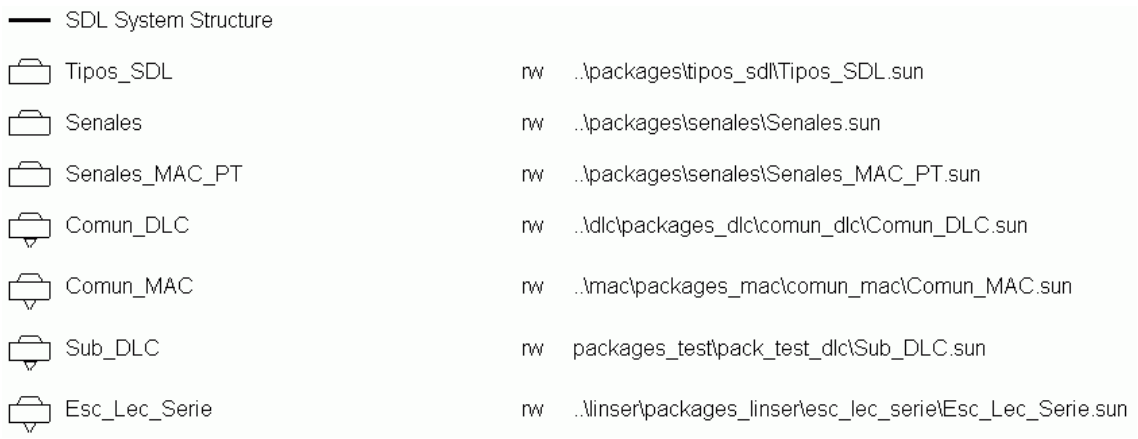


Figura 8.23: Paquetes incluidos en el Módulo de Protocolos `MProt_DLC_FT`.

8.5.2.2 Diseño detallado

En la etapa de Diseño Detallado, se realiza el modelado del comportamiento de cada uno de los procesos definidos en la actividad anterior. Cada Sistema de Pruebas incorpora un conjunto diferente de bloques, aunque algunos de estos bloques son utilizados en más de un Sistema de Pruebas. Esta información se presenta, de forma resumida, en la Tabla 8.29. La descripción del modelado de los procesos se ha incluido en el Apéndice J; en este apartado se describe, como ejemplo, el modelado del proceso `MBC_CTRL` de la Terminación Fija. El diseño de los procesos se ha realizado de forma que el número de estados de cada uno sea el mínimo.

Módulo de Protocolos, como por ejemplo paquetes con definiciones de señales o tipos de datos.

SPICUM servicio de publicaciones

Tabla 8.29: Procesos que constituyen los bloques incluidos en cada Sistema de Pruebas.

			Sistema de Pruebas		
Nivel	**Bloques**	**Procesos**	**SP_DLC_PT**	**SP_DLC_FT**	**SP_NWK_PT**
Acceso al Medio	MAC_CCF_FT	BMC	✓	-	✓
		MBC_CTRL			
		MBC			
		MBC_SELEC			
	MAC_CCF_PT	BMC	-	✓	-
		MBC_CTRL			
		MBC			
		MBC_SELEC			
	LINSER_FT	LINSER_FT	✓	-	✓
	LINSER_PT	LINSER_PT	-	✓	-
Control del Enlace	DLC_FT	CTRL_FT	-	-	✓
		LAPC_FT			
		Lc_FT			
		SignalROUTER			
		Lb_FT			
	SUB_DLC_FT	ConversorTTCN	✓	-	-
		Cuasi_Lc			
		Signal_RTX			
	SUB_DLC_PT	ConversorTTCN	-	✓	-
		Cuasi_Lc			
		Signal_RTX			
	AJUSTE_TIPOS_FT	AJUSTE_TIPOS_FT	-	-	✓
Gestión	LLME_FT	LLME_DLC_PT	-	-	✓
		LLME_MAC_PT			
	LLME_MAC_FT	LLME_MAC_FT	✓	-	-
	LLME_MAC_PT	LLME_MAC_PT	-	✓	-

8.5.2.2.1 Detalle del Modelado de un Proceso

El proceso MBC_CTRL es el encargado de la creación y el mantenimiento de las conexiones de tráfico. Si se autoriza la petición de conexión, confirma el establecimiento y crea una instancia del proceso MBC que se encargará de gestionar la nueva conexión; el proceso MBC_CTRL redirecciona a la instancia adecuada los mensajes del Nivel DLC. Durante una reinicialización del Subsistema Inferior, se encarga de la desconexión controlada del enlace activo.

```
/* Canal físico */
newtype CANAL
  struct
    slot Integer;          /* Intervalo temporal [0..11] */
    cn   Integer;          /* Número de portadora */
endnewtype CANAL;

/* Identificador de conexión */
newtype MBC_ID
  struct
    pid Pid := null;        /* PId de la instancia MBC asociada */
    mcei MCEI := -1;        /* Identificador de la conexión */
    canal CANAL;            /* Canal físico */
    pmid Bit_String_20;     /* Identidad de la Terminación Portátil*/
endnewtype MBC_ID;
```

Figura 8.24: Estructura de datos del proceso MBC_CTRL para almacenar la información de la conexión activa.

SPICUM servicio de publicaciones

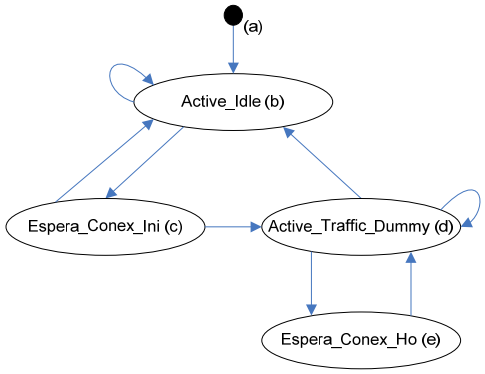
La información de la conexión activa se almacena en una variable de tipo `MBC_ID` (Figura 8.24), que contiene el PId de la instancia del proceso `MBC` asociada, el identificador de la conexión, el canal físico empleado y la identidad de la Terminación Portátil.

El diagrama de estados de este proceso y las transiciones posibles en cada uno de ellos se muestran en la Figura 8.25. El comportamiento de cada estado es el siguiente:

- `Active_Idle`: Es el estado inicial, en el cual no hay ninguna conexión activa y se está a la espera de una petición de conexión (`CSF_TBC_IND`) desde el Módulo de Capa Física. Cuando esta petición llega, y se autoriza, se crea una instancia del proceso `MBC` para gestionar la conexión.
- `Espera_Conex_Ini`: Espera la confirmación, procedente de la instancia del proceso `MBC` creada, de que la conexión se ha establecido correctamente. Si se ha producido algún error, se libera la posición asignada en la lista de conexiones.
- `Active_Traffic_Dummy`: Se llega a este estado cuando hay una conexión activa. En este estado se encaminan las señales recibidas desde el Nivel DLC hacia la instancia `MBC`. Pueden recibirse peticiones de envío de datos (`MAC_CO_DATA_REQ`), peticiones de desconexión del Nivel DLC (`MAC_DIS_REQ`) y peticiones de desconexión desde la Terminación Portátil (`MBC_DIS`). La indicación de traspaso se recibe con la misma primitiva de la petición de conexión (`CSF_TBC_IND`), indicando en el tipo que se trata de un traspaso; en este caso, se crea una segunda instancia del proceso `MBC` y se pasa al estado `Espera_Conex_Ho`.
- `Espera_Conex_Ho`: Espera la confirmación de que el traspaso se ha realizado con éxito.

	Estados				
Señales	a	b	c	d	e
-	b				
BMC_MBC_SINC_ERR		b		d	
CSF_TBC_IND		b,c		d,e	
MAC_CO_DATA_REQ				d	
MAC_DIS_REQ				d	
MBC_DIS			b	b,d	d
MBC_MBC_CTRL_CONEX			d		d

(a)



(b)

Figura 8.25: (a) Transiciones posibles y (b) Diagrama de estados para el proceso `MBC_CTRL` de la Terminación Fija.

8.5.2.3 Pruebas de Nivel

Las Pruebas de Nivel se han realizado para los Niveles DLC y MAC. Se han comprobado los bloques que componen los Módulos de Protocolos de cada Sistema de Pruebas (Sección 8.5.2.1), salvo los bloques `LINSER_FT`, `LINSER_PT` y `AJUSTE_TIPOS_FT`. Los bloques `LINSER` dependen de la plataforma donde se ejecuten,

SPICUM servicio de publicaciones

ya que realizan la comunicación con el Módulo de Capa Física, por lo que serán probados durante la fase de integración, concretamente, durante la integración con dicho Módulo. El tercer bloque mencionado es un mero traductor y no posee la entidad de Nivel.

Los sistemas que se han construido para realizar estas pruebas requieren el uso de niveles de protocolo que hasta ahora no se han implementado. Estos niveles se han modelado como bloques emuladores, ya que la funcionalidad que se ha incluido ha sido solamente la necesaria para la realización de las pruebas.

A continuación se describe la arquitectura de los sistemas empleados para realizar las Pruebas de Nivel, así como una breve descripción de los bloques emuladores que se han construido. Tras ello, se presentan ejemplos de la ejecución de estas pruebas.

8.5.2.3.1 Arquitecturas de las Pruebas de Nivel

Esta sección describe la estructura de los sistemas SDL utilizados en las Pruebas de Nivel. Las pruebas realizadas son las que se enumeran en la Tabla 8.25, la Tabla 8.26 y la Tabla 8.27.

Nivel MAC

Para las pruebas sobre el Nivel MAC (Tabla 8.25) se han enfrentado los niveles de las Terminaciones Fija y Portátil. Con el mismo conjunto de pruebas se ha comprobado también el comportamiento de los bloques de gestión de este nivel (`LLME_MAC_FT`, `LLME_MAC_PT`). La arquitectura del sistema SDL utilizado se muestra en la Figura 8.26, en la cual aparecen bloques emuladores para los niveles superior e inferior adyacentes del Nivel MAC.

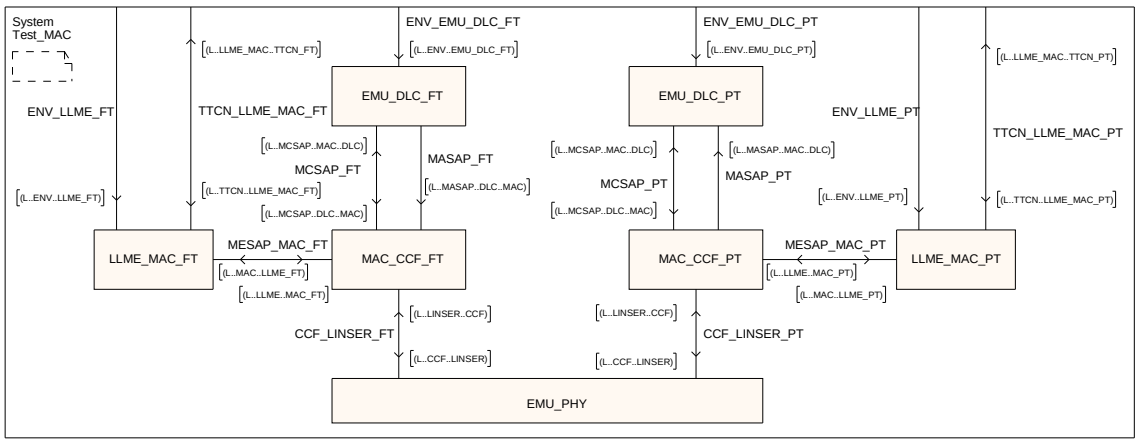


Figura 8.26: Arquitectura del sistema SDL empleado para las Pruebas del Nivel MAC.

Nivel DLC

Sobre el Nivel DLC se han ejecutado dos conjuntos de pruebas (Sección 8.5.1.3.2): una batería de pruebas básicas y los Juegos de Pruebas Abstractas. La batería de pruebas básicas se ha ejecutado sobre un sistema SDL que contiene tan sólo el correspondiente Nivel DLC, ya sea una Terminación Fija o una Portátil. Para ejecutar los Juegos de Pruebas se han utilizado las arquitecturas mostradas en la Figura 8.27. La arquitectura incluye emuladores de los niveles adyacentes (NWK y MAC), así como el bloque

SPICUM servicio de publicaciones

necesario para la comunicación con los Subsistemas de Pruebas (SUB_DLC_PT, SUB_DLC_FT). Al igual que en las Pruebas del Nivel MAC, el bloque de gestión se ha probado al mismo tiempo que el Nivel.

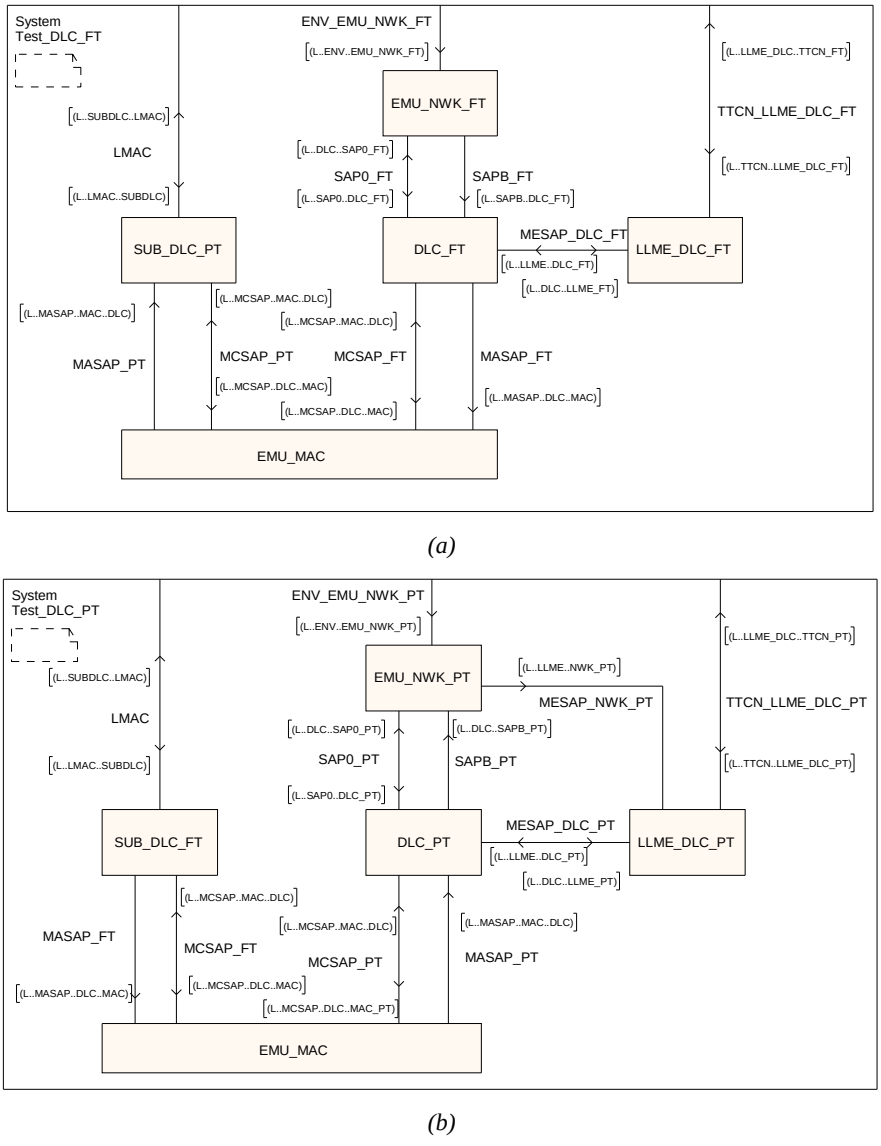


Figura 8.27: Arquitectura del sistema SDL empleado para las Pruebas del Nivel DLC de la (a) Terminación Fija y (b) Terminación Portátil.

8.5.2.3.2 Diseño de bloques emuladores

Los bloques emuladores incluyen la funcionalidad mínima imprescindible para realizar las pruebas. La mayor parte de esta funcionalidad consiste en responder automáticamente a las primitivas que recibe, pero también incorporan canales de comunicación con el exterior de forma que el operador pueda guiar el desarrollo de las pruebas. El diseño detallado de estos bloques se presenta en el Apéndice J, Sección J.4. Los emuladores que se han construido (Figura 8.26 y Figura 8.27) han sido los siguientes:

SPICUM servicio de publicaciones

- EMU_NWK_FT: Es el emulador del Nivel de Red de la Terminación Fija e incorpora la funcionalidad necesaria para poder realizar las pruebas del Juego de Pruebas SP_DLC_FT_Gap.mp sobre el Nivel DLC de dicha Terminación. Se comunica con el Nivel DLC a través de las interfaces SAPB_FT y SAP0_FT. El operador puede indicar el inicio de la transmisión en el canal de difusión.
- EMU_NWK_PT: Es el emulador del Nivel de Red de la Terminación Portátil. Incorpora la funcionalidad necesaria para poder realizar las pruebas del Juego de Pruebas SP_DLC_PT_Gap.mp sobre el Nivel DLC de dicha Terminación. Se comunica con el Nivel DLC a través de las interfaces SAPB_PT y SAP0_PT y con el bloque de gestión a través de la interfaz LLME_NWK. El operador puede controlar la operación de la Terminación Portátil con las acciones crear conexión (CREAR_CONEX), enviar datos (ENVIAR_DATOS) y liberar conexión (CERRAR_CONEX).
- EMU_DLC_FT: Emula el comportamiento del Nivel DLC de la Terminación Fija. Contiene un proceso para el envío de mensajes de difusión (EMU_DLC_BS_FT), y otro para gestionar las conexiones y el envío de datos (EMU_DLC_DATOS_FT).
- EMU_DLC_PT: Emula el comportamiento del Nivel DLC de la Terminación Portátil. Al igual que para la Terminación Fija, contiene dos procesos, uno para recibir los mensajes de difusión (EMU_DLC_BS_PT) y otro para gestionar las conexiones y recibir y enviar datos (EMU_DLC_DATOS_PT). Este emulador funciona de forma muy parecida al de la Terminación Fija; las principales diferencias son que a través del simulador se puede solicitar la creación de conexiones de tráfico y su traspaso, mientras que la Terminación Fija simplemente reaccionaba a estas solicitudes recibidas de la entidad remota.
- EMU_MAC: El bloque emulador del Nivel MAC contiene dos procesos, cada uno de los cuales emula la funcionalidad mínima correspondiente al Nivel MAC de cada Terminación Fija (EMU_MAC_FT) o Portátil (EMU_MAC_PT). En esencia, se trata de un 'tubo' que traduce las primitivas entre ambos extremos; cada proceso traduce las peticiones que recibe del Nivel DLC local a primitivas de indicación (creación de conexión[32], transferencia de datos, liberación de conexión y *paging*[33]).
- EMU_PHY: Emula el comportamiento del Módulo de Capa Física conjuntamente para ambas Terminaciones. Incorpora la funcionalidad mínima para comunicar una Terminación Fija con una Terminación Portátil; convierte las primitivas transmitidas por un extremo en las correspondientes a recibir en el otro.

8.5.2.3.3 Ejecución de las Pruebas de Nivel

Se han utilizado simuladores (Figura 8.29-b) para probar el comportamiento de los Niveles MAC de ambas Terminaciones y para realizar las pruebas básicas del comportamiento del DLC. Para realizar las pruebas con mayor comodidad, se han creado botones específicos dentro del simulador para controlar la operación de las mismas. Cada botón ejecuta una secuencia de acciones (un guión o *script*). Un ejemplo

[32] Sólo la Terminación Portátil.

[33] Sólo la Terminación Fija.

SPICUM servicio de publicaciones

se muestra en la Figura 8.29-a, donde aparecen los grupos de botones para interaccionar con el Nivel DLC de la Terminación Portátil.

Por ello, la comprobación del Nivel DLC mediante los Juegos de Pruebas estandarizados ha sido necesario realizarla fuera del entorno de simulación. Se ha generado una aplicación independiente a partir de cada sistema SDL, para lo que se han empleado los ejecutables obtenidos en la fase Construcción del Subsistema de Pruebas (Sección 8.4.2). La plataforma utilizada ha sido el sistema operativo Solaris 7.

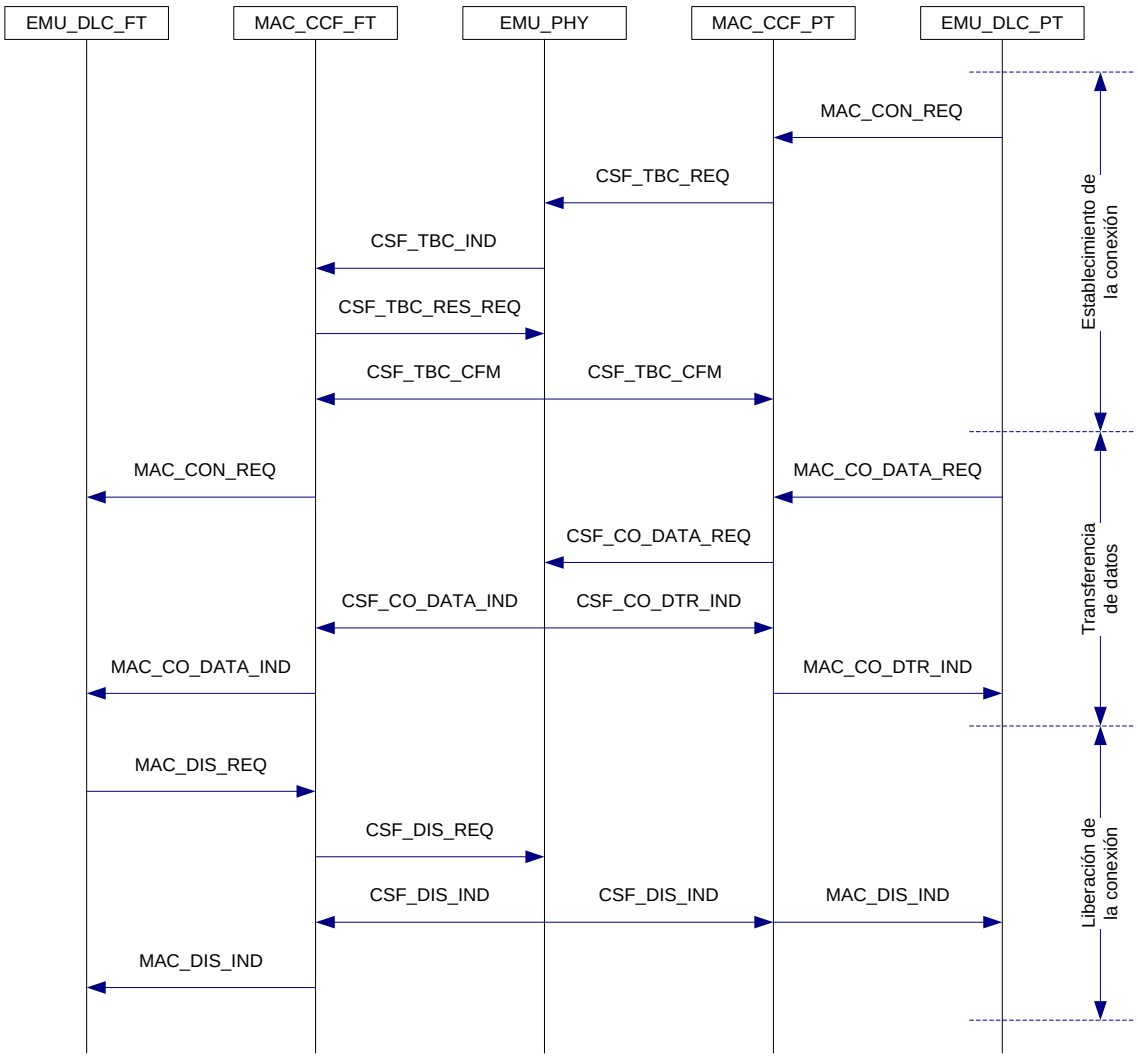


Figura 8.28: Secuencia de mensajes generada en las pruebas M1 (establecimiento), M2 (Transferencia) y M3 (liberación) del Nivel MAC.

Se muestran a continuación como ejemplo dos secuencias de mensaje obtenidas durante la realización de las Pruebas de Nivel. La Figura 8.28 muestra la interacción básica de mensajes de las pruebas de establecimiento de una conexión (M1), transferencia de datos (M2) y liberación de la conexión (M3) (ver Sección 8.5.1.3.2). La Figura 8.30 muestra la secuencia de mensajes, incluyendo el contenido de los mismos, generada en un establecimiento de la conexión iniciado por PT.

SPICUM servicio de publicaciones

(a)

Simuladores

```
Sim_MAC_FT.sct
Sim_MAC_PT.sct
Sim_DLC_PT.sct
Sim_DLC_FT.sct
```

(b)

Figura 8.29: (a) Botones de control de la operación del emulador de Nivel DLC de la Terminación Portátil y (b) Simuladores construidos para las Pruebas de Nivel.

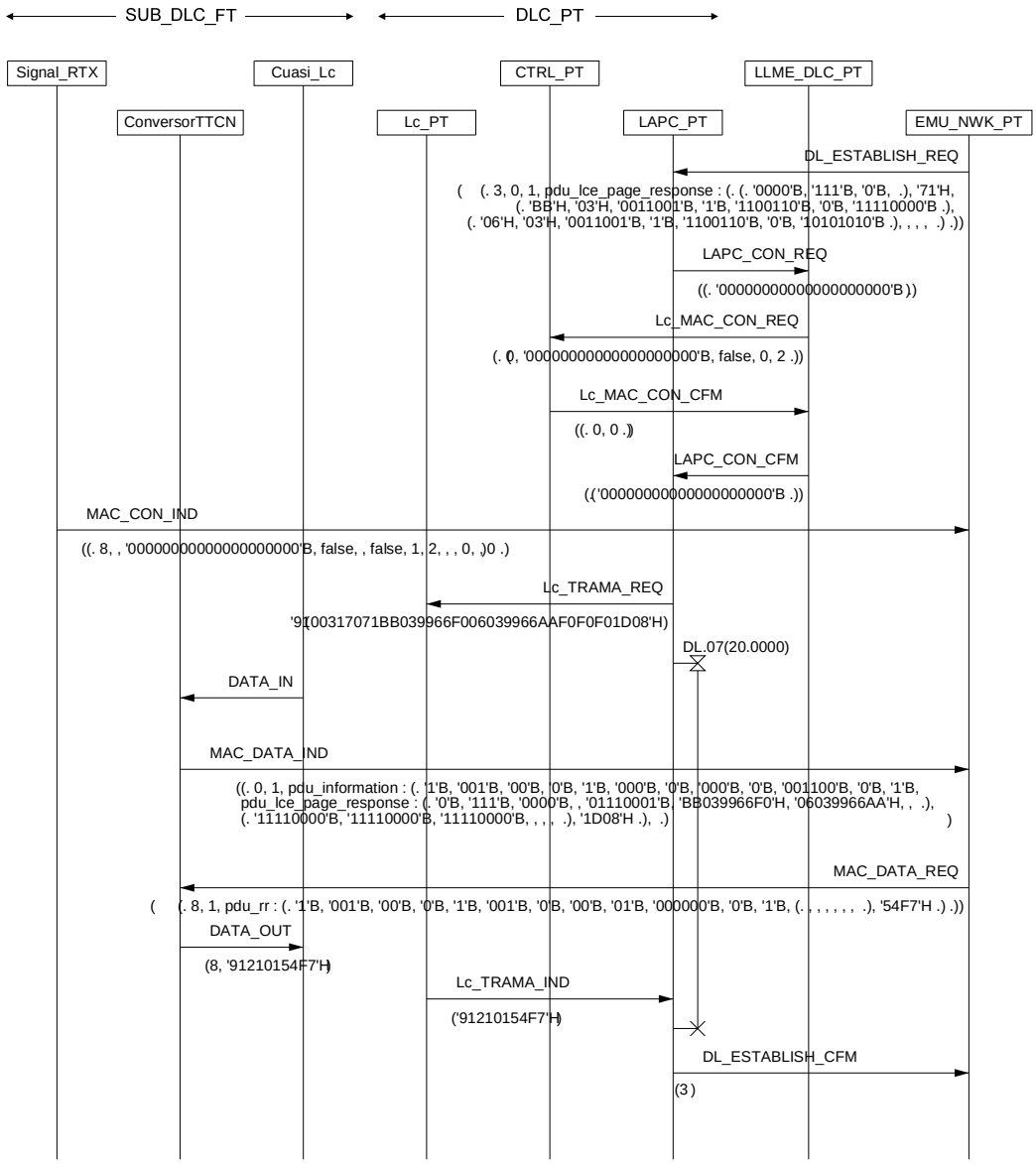


Figura 8.30: Secuencia de mensajes generada en la prueba de establecimiento de la conexión iniciado por PT.

SPICUM servicio de publicaciones

8.5.3 Integración del Subsistema Inferior

La integración de cada Subsistema Inferior se ha realizado integrando por separado el Módulo de Protocolos y el Módulo de Capa Física con el Nivel MAC. Una vez verificada la funcionalidad de estas integraciones se ha procedido a la integración del Subsistema. Para comprobar los Subsistemas Inferiores se han ejecutado, a través de ellos, las pruebas de conformidad de cada nivel, para lo que se han empleado emuladores de los Sistemas Bajo Prueba.

En primer lugar se explica la integración de los Módulos de Protocolos (Sección 8.5.3.1); posteriormente, se describe la integración del Módulo de Capa Física y el Nivel MAC (Sección 8.5.3.2). Finalmente se describe la obtención de una entidad ejecutable (Sección 8.5.3.3).

8.5.3.1 Integración del Módulo de Protocolos

Para los Sistemas de Pruebas DECT, la subfase de integración del Módulo de Protocolos es, en esencia, la integración de los bloques correspondientes al Nivel MAC y al Nivel DLC modelados en SDL. En el caso de los Sistemas de Pruebas para el Nivel DLC, la integración se realiza con el bloque `SUB_DLC_PT` o `SUB_DLC_FT`, mientras que para el Sistema de Pruebas para el Nivel NWK hay que integrar el Nivel DLC completo y el bloque `AJUSTE_TIPOS_FT` (ver Sección 8.5.1.1.2).

Tabla 8.30: Bloques incluidos en el sistema de integración de cada Módulo de Protocolos, agrupados por entidad funcional.

Sistema para la Integración del Módulo de Protocolos	**Sistema de Pruebas**	**Bloques**		
		Módulo de Protocolos ***(Nombre /Bloques)***	***Niveles de Protocolo del Emulador de Sistema Bajo Prueba*** ***(Nombre / Bloques)***	***Común***
`Integ_MProt_DLC_PT`	`SP_DLC_PT`	*`MProt_DLC_PT`*	*`EMU_Prot_SUT_DLC_PT`*	`EMU_PHY`
		`SUB_DLC_FT` `MAC_CCF_FT` `LLME_MAC_FT`	`EMU_NWK_PT` `DLC_PT` `MAC_CCF_PT` `LLME_PT`	
`Integ_MProt_DLC_FT`	`SP_DLC_FT`	*`MProt_DLC_FT`*	*`EMU_Prot_SUT_DLC_FT`*	
		`SUB_DLC_PT` `MAC_CCF_PT` `LLME_MAC_PT`	`EMU_NWK_FT` `DLC_FT` `MAC_CCF_FT` `LLME_FT`	
`Integ_MProt_NWK_PT`	`SP_NWK_PT`	*`MProt_NWK_PT`*	*`EMU_Prot_SUT_NWK_PT`*	
		`AJUSTE_TIPOS_FT` `DLC_FT` `MAC_CCF_FT` `LLME_FT`	`EMU_IWU_PT` `NWK_PT` `DLC_PT` `MAC_CCF_PT` `LLME_PT`	

Los bloques utilizados en cada integración se muestran en la Tabla 8.30. Se han agrupado según pertenezcan funcionalmente al Módulo de Protocolos o al emulador de

SPICUM servicio de publicaciones

Sistema Bajo Prueba. Se ha denominado `EMU_Prot_SUT_xxxx` a la torre de protocolos que corresponde al emulador del Sistema Bajo Prueba. Para comprobar las integraciones se han utilizado las Pruebas de Módulo definidas en la Sección 8.5.1.3.3. Los sistemas SDL construidos para ello son los siguientes:

- `Integ_MProt_DLC_PT`: Incluye el Módulo de Protocolos (`MProt_DLC_PT`) para el Sistema de Pruebas `SP_DLC_PT` y un emulador de los niveles de protocolo del SUT (`EMU_Prot_SUT_DLC_PT`) hasta el Nivel de Red de la Terminación Portátil.
- `Integ_MProt_DLC_FT`: Incluye el Módulo de Protocolos (`MProt_DLC_FT`) para el Sistema de Pruebas `SP_DLC_FT` y un emulador de los niveles de protocolo del SUT (`EMU_Prot_SUT_DLC_FT`) hasta el Nivel de Red de la Terminación Fija.
- `Integ_MProt_NWK_PT`: Incluye el Módulo de Protocolos (`MProt_NWK_PT`) para el Sistema de Pruebas `SP_NWK_PT` y un emulador de los niveles de protocolo del SUT (`EMU_Prot_SUT_NWK_PT`) hasta el Nivel de Aplicación (IWU – *InterWorking Unit*) de la Terminación Portátil.

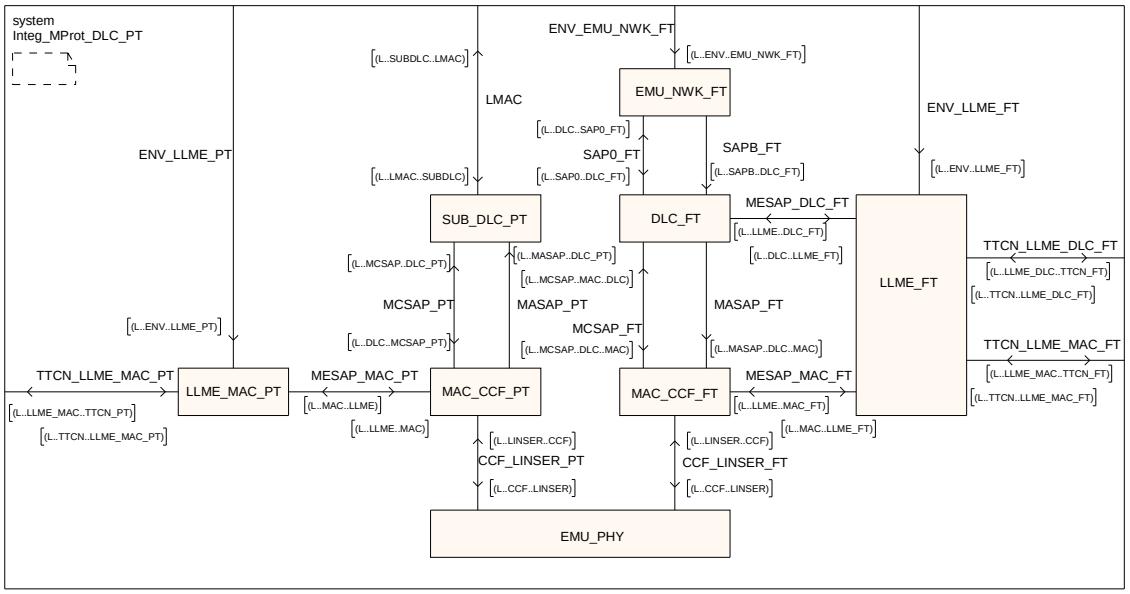


Figura 8.31: Integración del Módulo de Protocolos del Sistema de Pruebas `SP_DLC_FT` (Terminación Fija).

La integración de los Módulos de Protocolos de los Sistemas de Pruebas para el Nivel DLC, `Integ_MProt_DLC_PT` e `Integ_MProt_DLC_FT`, se ha realizado como se muestra en la Figura 8.31. El Módulo de Protocolos del Sistema de Pruebas para el Nivel NWK (`MProt_NWK_PT`) se ha integrado, como se muestra en la Figura 8.32, en el sistema `Integ_MProt_NWK_PT`. Este sistema SDL incorpora dos nuevos bloques que son necesarios para la ejecución de las pruebas: el bloque `NWK_PT`, que modela el Nivel de Red de la Terminación Portátil, y el bloque `EMU_IWU_PT`, que emula la funcionalidad por encima del Nivel de Red. La implementación del Nivel de Red se describe en el Apéndice K; el bloque `EMU_IWU_PT` se describe en el Apéndice J, Sección J.4.1.

SPICUM servicio de publicaciones

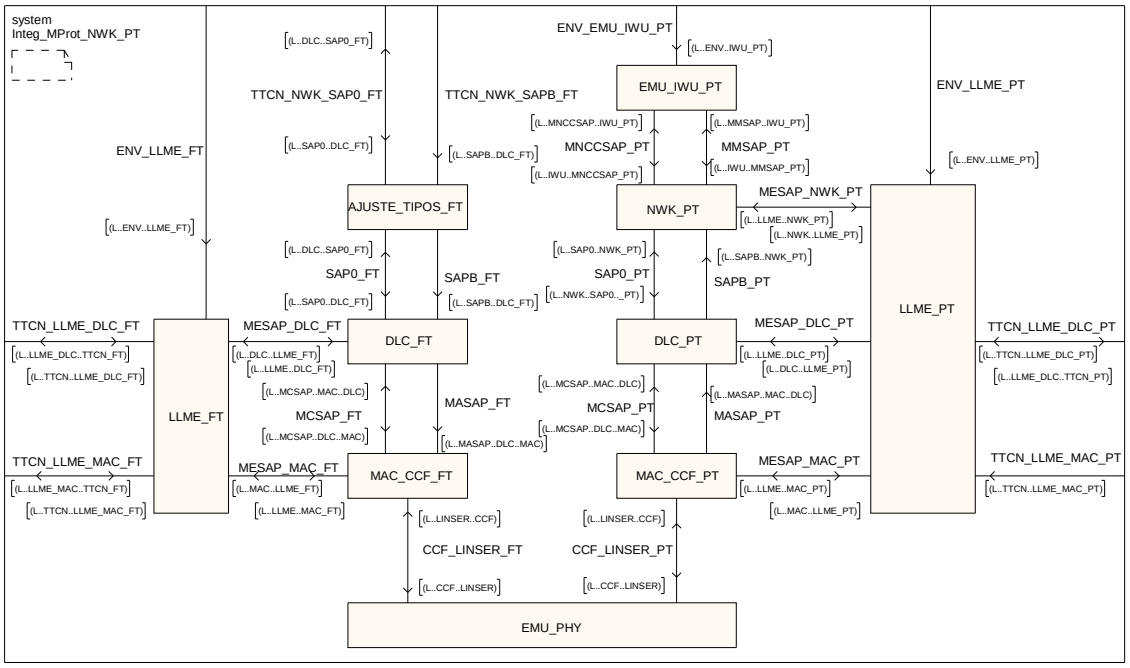


Figura 8.32: Integración del Módulo de Protocolos del Sistema de Pruebas SP_NWK_PT (Terminación Portátil).

8.5.3.1.1 Pruebas de Módulo

Los Módulos de Protocolos se han probado en un sistema SDL con un emulador del Sistema Bajo Prueba; la comunicación entre ambos se realiza mediante un emulador del Módulo de Capa Física. Las pruebas son las indicadas en la Sección 8.5.1.3.3.

8.5.3.2 Integración del Módulo de Capa Física

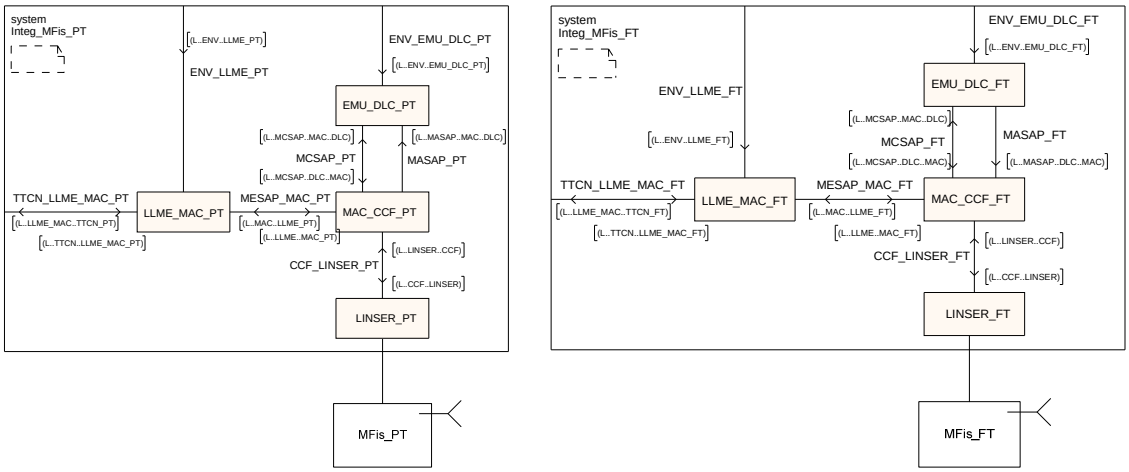
La integración del Módulo de Capa Física con el Nivel MAC se ha realizado sobre el sistema mostrado en la Figura 8.26. El emulador de Nivel Físico (`EMU_PHY`) ha sido sustituido por los bloques `LINSER` y el módulo SiTel Sierra (ver Sección 8.5.1.1) de cada Terminación. La comunicación entre el modelo SDL y el Módulo de Capa Física se realiza a través de un manejador desarrollado en C; el diseño de este manejador se explica en la Sección 8.5.3.2.1.

La inclusión del Módulo de Capa Física en la arquitectura de bloques hace que el sistema mencionado se divida en dos, uno correspondiente a la Terminación Fija, `Integ_MFis_FT` (Figura 8.33-a), y otro a la Terminación Portátil, `Integ_MFis_PT` (Figura 8.33-b). La comprobación de la integración se ha realizado enfrentando las dos Terminaciones, cada una en su respectivo simulador, y ejecutando las mismas pruebas que para comprobar el Nivel MAC; la comunicación entre ambas Terminaciones se lleva a cabo vía radio. Dado que los Módulos de Capa Física disponen de su propio reloj, los simuladores que se han generado incorporan el motor de tiempo real.

Una vez integrado el Módulo de Capa Física con el Nivel MAC se ha procedido a la integración de los Módulos de Protocolos con el Módulo de Capa Física. Se obtienen así los Subsistemas Inferiores correspondientes a cada uno de los Sistemas de Pruebas indicados en la Definición de la Estructura (ver Sección 8.5.1.1.2, Tabla 8.12). Los Módulos que componen cada uno de ellos se listan en la Tabla 8.31. La Figura 8.34 muestra el Subsistema Inferior empleado en las pruebas el Nivel DLC de la

SPICUM servicio de publicaciones

Terminación Portátil. Para comprobar la integración se han empleado los emuladores de los Sistemas Bajo Prueba descritos en la Sección 8.5.3.2.2.



(a) *(b)*

Figura 8.33: Modelo SDL de la integración del Módulo de Capa Física con el Nivel MAC de la Terminación (a) Fija y (b) Portátil.

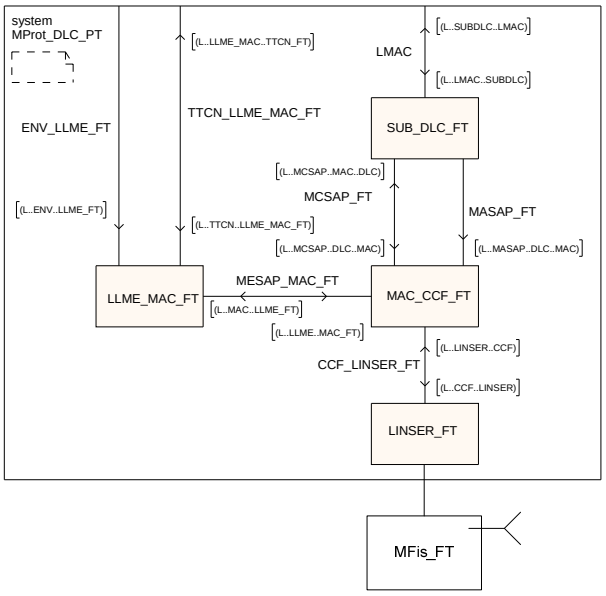


Figura 8.34: Subsistema Inferior para el Sistema de Pruebas del Nivel DLC de la Terminación Portátil.

SPICUM servicio de publicaciones

Tabla 8.31: Módulos que componen cada Subsistema Inferior y Sistema de Pruebas al que pertenecen estos.

Subsistema Inferior	Sistema de Pruebas	Módulo de Protocolos		Módulo de Capa Física
		Nombre	Bloques	
SI_DLC_PT	SP_DLC_PT	*MProt_DLC_PT*	SUB_DLC_FT MAC_CCF_FT LLME_MAC_FT LINSER_FT	MFis_FT
SI_DLC_FT	SP_DLC_FT	*MProt_DLC_FT*	SUB_DLC_PT MAC_CCF_PT LLME_MAC_PT LINSER_PT	MFis_PT
SI_NWK_PT	SP_NWK_PT	*MProt_NWK_PT*	AJUSTE_TIPOS_FT DLC_FT MAC_CCF_FT LLME_FT LINSER_FT	MFis_FT

8.5.3.2.1 Manejador del Módulo de Capa Física

La interfaz de los Módulos de Capa Física es una línea serie V.24, con control de flujo por *software* y una velocidad máxima de 1200 baudios. El manejador permite controlar esta interfaz desde un PC (*Personal Computer*) o estación de trabajo y ha sido diseñado como tres hilos: el principal, uno de lectura y otro de escritura. La comunicación con el nivel superior se realiza a través del hilo principal, que permite acceder la cola de lectura y la de escritura. Los otros dos hilos controlan la línea serie y realizan la verdadera comunicación con los Módulos de Capa Física.

Tabla 8.32: Interfaz de funciones ofrecido por el manejador.

Función	Descripción
abrir_csf	Realiza la apertura del '*canal*' asociado al puerto serie
cerrar_csf	Realiza el cierre del '*canal*' asociado al puerto serie
configurar	Configura parámetros del puerto serie
reset	Genera un pulso (pin DTR[34]) para resetear los módulos
leer_csf	Lee una primitiva de la cola de recepción
escribir_csf	Pone una primitiva en la cola de transmisión
driver_sitel	Lanza la ejecución de los hilos de lectura y escritura del manejador
kill_threads	Termina los hilos de ejecución de lectura y escritura
set_timer_dl07	Activa el temporizador dl07 para trazas de medida de tiempos
reset_timer_dl07	Detiene el temporizador dl07 y muestra en pantalla su duración

SPICUM servicio de publicaciones

El manejador realiza la comunicación con el Módulo de Capa Física a través de la interfaz de funciones (`driver_sitel.h`) mostrada en la Tabla 8.32. La codificación y descodificación de la información se realiza en el bloque `LINSER`; se describe en el Apéndice J, Sección J.1.1.5. El código ha sido escrito siguiendo el estándar POSIX [POSI99]. La Figura 8.35 muestra una traza de las primitivas intercambiadas durante el procedimiento de enganche y creación de un canal de la Terminación Portátil en la línea serie; en negrita aparecen los mensajes descendentes. Al estar escrito en C, el manejador no se puede depurar con el simulador de la herramienta SDL, por eso las trazas se generan a través de la salida estándar de error, `stderr`. La generación o no de estas trazas se controla mediante variables de compilación condicional.

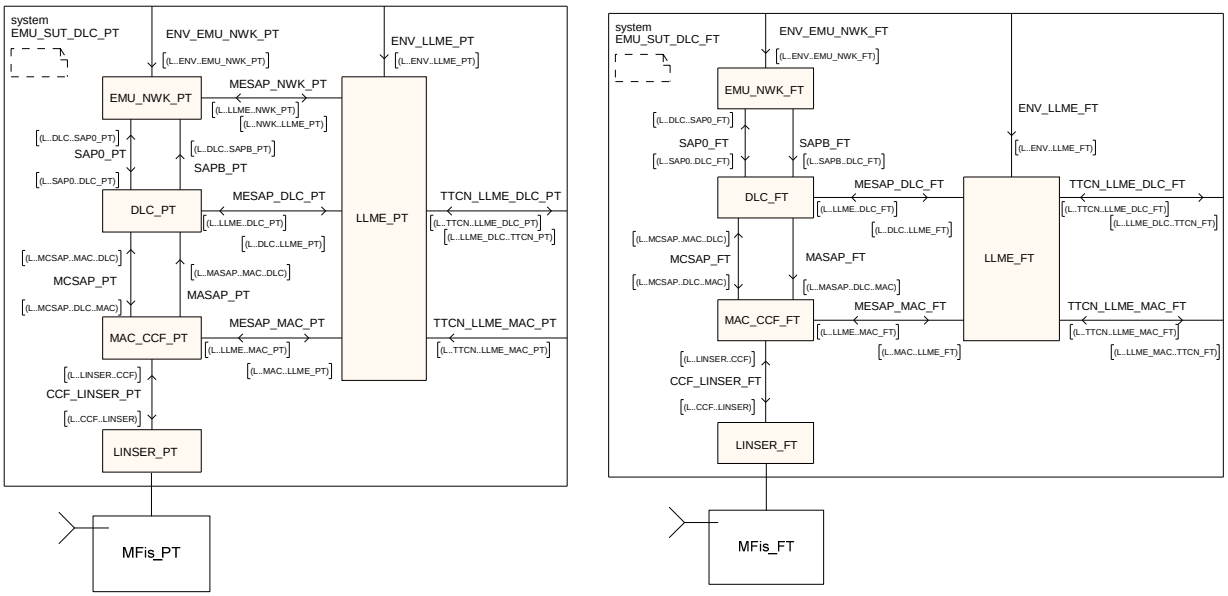
```
SCAN_REQ 1

00 CSF_NT           01 02 03 04 01
00 CSF_FP_CAP       32 03 04 0B 0B
03 CSF_STAT_INFO    03 00 00 01 04

DBC_REQ 3 : 1

00 CSF_NT           01 02 03 04 01
03 CSF_STAT_INFO    03 00 00 01 04
03 CSF_FP_CAP       32 03 04 0B 0B
03 CSF_PAGE_IND     10 50 03 01 00 23 45

PP_PAGE_INFO_SHORT_IND   03 : 10 : 50 : 03 : 01 : 00

TBC_REQ 3

03 CSF_TBC_CFM

CO_DATA_REQ 3

03 CSF_CO_DTR_IND
```

Figura 8.35: Primitivas intercambiadas en la línea serie de la Terminación Portátil durante el procedimiento de enganche y creación de un canal.

8.5.3.2.2 Emuladores de Sistemas Bajo Prueba

Se han construido emuladores de Sistemas Bajo Prueba para poder comprobar el funcionamiento de los Sistemas de Pruebas. Su arquitectura se muestra en la Figura 8.36-a, la Figura 8.36-b y la Figura 8.37. Estos emuladores se han ido integrando al mismo tiempo que los Módulos de Protocolos, como se puede observar comparando estas figuras con la Figura 8.31 y la Figura 8.32. En estas últimas, mientras la mitad izquierda corresponde al Módulo de Protocolos, la otra torre corresponde al emulador del Sistema Bajo Prueba.

SPICUM servicio de publicaciones



(a) *(b)*

Figura 8.36: Emuladores de Sistemas Bajo Prueba para los Sistemas de Pruebas del Nivel DLC de (a) la Terminación Portátil (SP_DLC_PT) y (b) de la Terminación Fija (SP_DLC_FT).

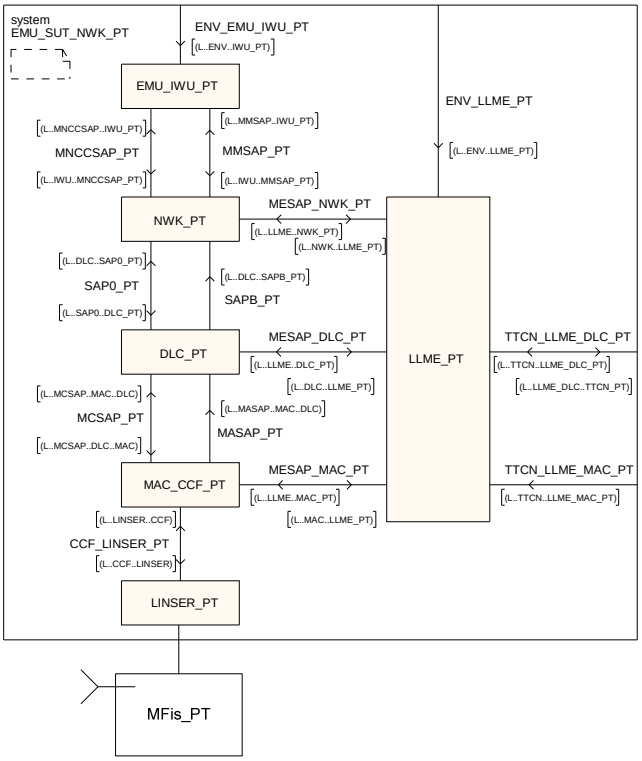


Figura 8.37: Emulador de Sistema Bajo Prueba para el Sistema de Pruebas del Nivel NWK de la Terminación Portátil (SP_NWK_PT).

SPICUM servicio de publicaciones

Tabla 8.33: Módulos que componen cada Emulador de Sistema Bajo Prueba y Sistema de Pruebas al que corresponden.

Emulador de Sistema Bajo Prueba	Sistema de Pruebas	Niveles de Protocolo		Módulo de Capa Física
		Nombre	Bloques	
EMU_SUT_DLC_PT	SP_DLC_PT	*EMU_Prot_SUT_DLC_PT*	EMU_NWK_PT DLC_PT MAC_CCF_PT LLME_PT LINSER_PT	MFis_PT
EMU_SUT_DLC_FT	SP_DLC_FT	*EMU_Prot_SUT_DLC_FT*	EMU_NWK_FT DLC_FT MAC_CCF_FT LLME_FT LINSER_FT	MFis_FT
EMU_SUT_NWK_PT	SP_NWK_PT	*EMU_Prot_SUT_NWK_PT*	EMU_IWU_PT NWK_PT DLC_PT MAC_CCF_PT LLME_PT LINSER_PT	MFis_PT

Los emuladores de Sistema Bajo Prueba que se han construido y los bloques que los constituyen se enumeran en la Tabla 8.33. Estos emuladores son los siguientes:

- EMU_SUT_DLC_FT (Figura 8.36-b): Implementa una Terminación Fija hasta el Nivel DLC. Los niveles superiores se emulan en el bloque EMU_NWK_FT. Se ha utilizado para probar el Sistema de Pruebas del Nivel DLC de la Terminación Fija.
- EMU_SUT_DLC_PT (Figura 8.36-a): Implementa una Terminación Portátil hasta el Nivel DLC. Los niveles superiores se emulan en el bloque EMU_NWK_PT. Se ha utilizado para probar el Sistema de Pruebas del Nivel DLC de la Terminación Portátil. Se podría emplear también el emulador EMU_SUT_NWK_PT, ya que incorpora la funcionalidad del Nivel DLC, pero de esta forma se controla mejor la operación durante las pruebas.
- EMU_SUT_NWK_PT (Figura 8.37): Implementa una Terminación Portátil hasta el Nivel de Red. Los niveles superiores se emulan en el bloque EMU_IWU_PT. Se ha utilizado para probar el Sistema de Pruebas del Nivel de Red de la Terminación Portátil.

Los bloques utilizados en estos emuladores son los de apartados anteriores, a los que se añaden los bloques NWK_PT, LLME_PT y EMU_IWU_PT. Los dos primeros, relacionados con el Nivel de Red de la Terminación Portátil, se describen en el Apéndice K[35]; el último se describe en el Apéndice J.

8.5.3.2.3 Pruebas de Subsistema

Para probar la integración del Subsistema Inferior se han ejecutado los mismos conjuntos de pruebas de conformidad que en las Pruebas de Módulo, pero ahora el emulador del Sistema Bajo Prueba constituye una entidad independiente (Figura 8.38). El Módulo de Protocolos se ejecuta como una aplicación[36] de la plataforma (su generación se explica en la Sección siguiente); el emulador del Sistema Bajo Prueba se

SPICUM servicio de publicaciones

ha ejecutado dentro del simulador. Se ha utilizado el motor de tiempo real para cumplir las restricciones temporales de la comunicación.

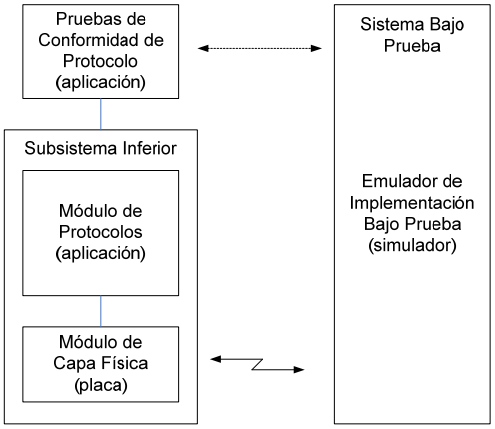


Figura 8.38: Esquema de la realización de las Pruebas de Subsistema.

8.5.3.3 Obtención de una Entidad Ejecutable del Subsistema Inferior

A partir del modelo SDL de cada Módulo de Protocolos se ha generado código para el compilador *gcc*. Este código se ha enlazado con (Tabla 8.34) el Motor TTCN de aplicación de tiempo real[37], el Módulo Adaptador de los Protocolos (Capítulo 5, Sección 5.4.1) y el Módulo de Gestión de los Protocolos (Capítulo 5, Sección 5.4.2).

Tabla 8.34: Ficheros necesarios para generar un ejecutable del Módulo de Protocolos del Sistema de Pruebas SP_DLC_PT.

Ficheros Generados	Librerías	Módulos Adaptador y de Gestión

`SP_DLC_PT.c`
`Comun_DLC.c`
`Sub_DLC.c`
`Comun_MAC.c`
`Senales.c`
`Senales_FT.c`
`Tipos_SDL.c`
`Esc_Lec_Serie.c`
	`Motor SDL`	`Entorno.c`
		`Codec_SDL.c`
		`Driver_SiTel_FT.c`

Los codificadores incluidos en el Módulo de Gestión de los Protocolos han tenido que ser construidos. El codificador de la comunicación con el Módulo de Capa Física, Codec $Local_{PHY}$, se ha descrito en la Sección 8.5.3.2.1. El codificador de la comunicación con el Subsistema de Pruebas, Codec $Local_{TTCN}$, se ha generado automáticamente con la herramienta *GenCod* (Capítulo 5, Sección 5.5.2.2) y utiliza la sintaxis de transferencia ASCII; este codificador es específico para cada Sistema de Pruebas. Como entradas a la herramienta se han utilizado las salidas obtenidas por la herramienta *GenDef* (Sección 8.5.1.2.1): ficheros de ASPs, PDUs y tipos de datos generados. El resultado son las funciones de codificación y descodificación incluidas en el fichero `SDL_Codec.c` pero requiere un par de retoques manuales:

SPICUM servicio de publicaciones

a) Actualización de los nombres de algunos tipos (ver Sección 8.5.1.2.1): se les ha añadido un sufijo por duplicidad; por ejemplo, la PDU `PRESENT_PDU_CC_SETUP`.

b) Inclusión de las primitivas de control que no están definidas en las pruebas (Sección 8.5.1.2.4). Un ejemplo del código necesario para estas señales se muestra en la Figura 8.39.

```
else if (strncmp(tipo, "CONFIG_FT", 9 ) ==0) {
  SP = xGetSignal(CONFIG_FT, xNotDefPId, xEnv);
  SDL_Output(SP xSigPrioPar(xDefaultSignalPrio), (xIdNode *)0);
  fprintf(stderr,"\nSeñal CONFIG_FT enviada al SDL...");
}
```

Figura 8.39: Código ejemplo para las señales de las interfaces de control.

8.6 Construcción del Sistema de Pruebas

La fase de construcción de los Sistemas de Pruebas consiste, básicamente, en poner en comunicación los Subsistemas que lo componen (Tabla 8.35), cada uno de los cuales es una entidad independiente:

Tabla 8.35: Subsistemas que componen cada Sistema de Pruebas.

Sistema de Pruebas	Subsistema de Operación y Administración	Subsistema de Pruebas	Subsistema Inferior
SP_DLC_PT	SubOpAd.java	Ets_dlc_pt.exe	SI_DLC_PT.exe
SP_DLC_FT		Ets_dlc_ft.exe	SI_DLC_FT.exe
SP_NWK_PT		Ets_nwk_pt.exe	SI_NWK_PT.exe

El Subsistema de Operación y Administración requiere dos ficheros de configuración:

- Configuración de los directorios de trabajo (ejecutables y trazas generadas).
- Configuración de los Juegos de Pruebas (Figura 8.40): indica el fichero de Parámetros de Pruebas, la entidad ejecutable del Subsistema Inferior y los canales de comunicación (nombre, dirección IP y número de puerto) a usar. La Figura 8.41 muestra un ejemplo de fichero de Parámetros de Pruebas.

Los Sistemas de Pruebas han sido construidos y probados sobre las plataformas Linux, Unix y Windows, sin más que recompilar el código de cada Módulo en la plataforma objetivo. Además, el Sistema de Pruebas del Nivel DLC de la Terminación Portátil, `SP_DLC_PT`, ha sido también construido sobre una plataforma con sistema operativo DSP/BIOS; esta adaptación se describe en la Sección 8.6.2.

SPICUM servicio de publicaciones

```
Pixit = /dect/GUI/nwk_pt/pixit_nwk_pt.ttp
Subsistema = /dect/GUI/nwk_pt/SI_NWK_PT.sub.exe
NumeroPCOs = 2
PCO_0_nombre = DLB
PCO_0_IP = 127.0.0.1
PCO_0_puerto_TCP = 8000
PCO_1_nombre = DLS
PCO_1_IP = 127.0.0.1
PCO_1_puerto_TCP = 8001
```

Figura 8.40: Configuración de las pruebas para el Sistema de Pruebas SP_NWK_PT.

```
<params-file>
TSPC_Lb                          BOOLEAN                    TRUE
TSPC_Lb_short_frame              BOOLEAN                    TRUE
TSPC_ca_info_transfer            BOOLEAN                    TRUE
TSPC_class0                      BOOLEAN                    TRUE
TSPC_classA                      BOOLEAN                    TRUE
TSPC_fu1                         BOOLEAN                    TRUE
TSPC_intercell_ho                BOOLEAN                    TRUE
TSPC_intracell_ho                BOOLEAN                    TRUE
TSPX_ari                         ARI
                                '01000100100010000011001110101010 1001'B
TSPX_ca_accept_est               BOOLEAN                    TRUE
TSPX_chn                         BOOLEAN                    FALSE
TSPX_decay_rate                  INTEGER                    5
TSPX_dl04_value                  INTEGER                    20000
TSPX_dl07_value                  INTEGER                    20000
TSPX_dummy_bearer_duration       INTEGER                    10
TSPX_fid                         FIXED_IDENTITY       '0606A0A015533333'O
TSPX_fu1_snd_pr_defined          BOOLEAN                    TRUE
TSPX_in_pdu                      OCTETSTRING                '02468ACE'O
TSPX_in_rec_proc_defined         BOOLEAN                    FALSE
TSPX_intracell_behaviour         INTEGER                    0
TSPX_ipui                        BITSTRING       '000011100010001101000101'B
TSPX_lbs_proc_defined            BOOLEAN                    TRUE
TSPX_n250                        INTEGER                    3
TSPX_pid                         PORTABLE_IDENTITY    '050780A80A8004E190'O
TSPX_pmid_assigned               PMID        '11010000111000010010001101 00'B
TSPX_pt                          BOOLEAN                    FALSE
TSPX_rpn                         RPN                        0
TSPX_rpn1                        RPN                        0
TSPX_slot                        SLOT_TYPE                  1
<params-file>
```

Figura 8.41: Ejemplo de fichero de Parámetros de Pruebas para Sistema de Pruebas del Nivel DLC de la Terminación Fija.

8.6.1.1 Pruebas de Sistema

Para comprobar el funcionamiento de los Sistemas de Pruebas se han utilizado las pruebas indicadas en la Sección 8.5.1.3.5. El emulador de la Implementación Bajo Prueba se puede ejecutar en modo simulación o como aplicación; para el segundo caso, se ha incorporado una interfaz de usuario que permite controlar su operación [SANC00a]. La distribución entre plataformas de cada elemento involucrado en la realización de estas pruebas se muestra en la Figura 8.42. Los Casos de Prueba que se han comprobado en cada Sistema de Pruebas se listan en la Tabla 8.36 y la Tabla 8.37.

SPICUM servicio de publicaciones

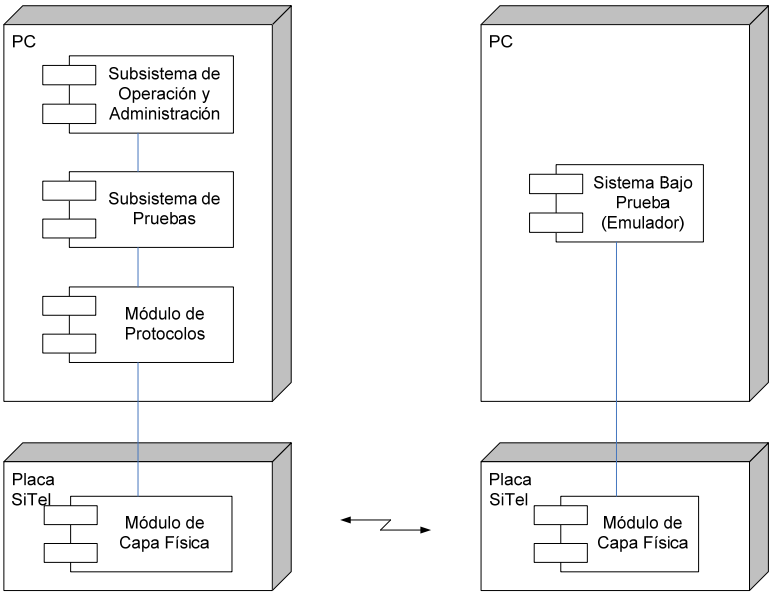


Figura 8.42: Ubicación de los componentes utilizados para realizar las Pruebas de Sistema.

Tabla 8.36: Lista de Casos de Prueba incluidos en los Sistemas de Pruebas de las Terminaciones Portátil y Fija del Nivel DLC.

Caso de Prueba	PT	FT	Caso de Prueba	PT	FT	Caso de Prueba	PT	FT
TC_A_CA_000	X	---	TC_A_BI_000	X	---	TC_A_BI_011	X	X
TC_A_CA_001	X	---	TC_A_BI_001	X	---	TC_A_BI_012	X	X
TC_A_CA_005	X	X	TC_A_BI_002	X	---	TC_A_BI_013	X	X
TC_A_CA_006	X	X	TC_A_BI_003	X	---	TC_A_BO_000	X	---
TC_A_CA_007	---	X	TC_A_BI_004	X	X	TC_A_BO_001	X	---
TC_A_CA_008	---	X	TC_A_BI_005	X	X	TC_A_BO_002	X	---
TC_A_BV_002	X	X	TC_A_BI_006	X	X	TC_A_BO_003	X	---
TC_A_BV_003	X	X	TC_A_BI_007	X	X	TC_L_CA_000	X	X
TC_A_BV_005	X	X	TC_A_BI_008	X	X	TC_0_CA_000	X	X
TC_A_BV_006	X	X	TC_A_BI_009	X	X	TC_0_CA_001	X	X

8.6.2 Adaptación a la Plataforma DSP/BIOS

El Sistema de Pruebas del Nivel DLC de la Terminación Portátil, SP_DLC_PT, ha sido también adaptado para una plataforma con sistema operativo DSP/BIOS [TIBIOS] con objeto de demostrar el uso de plataformas de tiempo real y la flexibilidad de la arquitectura [PLAZ06]. La asignación de componentes es la mostrada en la Figura 8.43. El Módulo de Protocolos se ejecuta sobre una tarjeta específica, la tarjeta EVM6701 [TIEVM], que incorpora un DSP (*Digital Signal Processor*) tipo TMS320C6701 [TIC67][38]. Con esta distribución se logra cumplir las constricciones temporales con mayor facilidad, ya que el acoplamiento entre el Módulo de Protocolos y el Módulo de Capa Física es mayor al ejecutarse el primero sobre una plataforma de tiempo real. Además, el PC queda más descargado, pudiendo realizar otro tipo de procesamientos

SPICUM servicio de publicaciones

durante la ejecución de una Prueba, por ejemplo, visualización en tiempo real de su evolución.

Tabla 8.37: Lista de Casos de Prueba incluidos en el Sistema de Pruebas de la Terminación Portátil del Nivel NWK.

Caso de Prueba	Caso de Prueba	Caso de Prueba
TC_PT_CC_BV_OC_01	TC_PT_CC_BV_CR_07	TC_PT_MM_BV_AR_03
TC_PT_CC_BV_OC_02	TC_PT_CC_BV_CR_08	TC_PT_MM_BV_AR_05
TC_PT_CC_BV_OC_03	TC_PT_CC_BV_CR_09	TC_PT_MM_BV_CH_01
TC_PT_CC_BV_OC_04	TC_PT_CC_BV_CR_10	TC_PT_MM_BV_CH_02
TC_PT_CC_BV_IC_01	TC_PT_CC_BV_CR_11	TC_PT_MM_BV_CH_05
TC_PT_CC_BV_IC_02	TC_PT_CC_BV_RS_01	TC_PT_MM_BO_01
TC_PT_CC_BV_CI_01	TC_PT_CC_BO_01	TC_PT_MM_BI_01
TC_PT_CC_BV_CI_02	TC_PT_CC_BO_02	TC_PT_MM_BI_02
TC_PT_CC_BV_CI_03	TC_PT_CC_BI_01	TC_PT_MM_BI_03
TC_PT_CC_BV_CI_04	TC_PT_CC_BI_02	TC_PT_MM_BI_04
TC_PT_CC_BV_CI_05	TC_PT_CC_BI_03	TC_PT_MM_TI_01
TC_PT_CC_BV_CI_06	TC_PT_CC_TI_01	TC_PT_MM_TI_03
TC_PT_CC_BV_CI_07	TC_PT_CC_TI_02	TC_PT_MM_TI_04
TC_PT_CC_BV_CI_08	TC_PT_CC_TI_03	TC_PT_MM_TI_05
TC_PT_CC_BV_CI_09	TC_PT_CC_TI_04	TC_PT_ME_BV_10
TC_PT_CC_BV_CI_10	TC_PT_MM_BV_ID_01	TC_PT_ME_BV_11
TC_PT_CC_BV_CI_11	TC_PT_MM_BV_ID_02	TC_PT_ME_BV_12
TC_PT_CC_BV_CI_12	TC_PT_MM_BV_ID_08	TC_PT_ME_BV_13
TC_PT_CC_BV_CI_13	TC_PT_MM_BV_AU_02	TC_PT_LC_BV_LE_01
TC_PT_CC_BV_CI_14	TC_PT_MM_BV_LO_01	TC_PT_LC_BV_LE_02
TC_PT_CC_BV_CR_01	TC_PT_MM_BV_LO_02	TC_PT_LC_BV_LR_01
TC_PT_CC_BV_CR_02	TC_PT_MM_BV_LO_03	TC_PT_LC_BV_LR_02
TC_PT_CC_BV_CR_03	TC_PT_MM_BV_LO_04	TC_PT_LC_BV_LR_03
TC_PT_CC_BV_CR_04	TC_PT_MM_BV_LO_05	TC_PT_LC_BI_01
TC_PT_CC_BV_CR_05	TC_PT_MM_BV_LO_06	TC_PT_LC_BI_03
TC_PT_CC_BV_CR_06	TC_PT_MM_BV_LO_07	TC_PT_LC_TI_02

El sistema operativo DSP/BIOS es un sistema de tiempo real multitarea con prioridades con desalojo. Las tareas se comunican entre sí a través de buzones. La comunicación con el PC anfitrión se realiza a través de la tecnología RTDX (*Real-Time Data Exchange*), que proporciona canales unidireccionales; esta comunicación se realiza en tiempo real sólo en el sentido PC → DSP, ya que la solicitud debe partir siempre del DSP.

El código generado a partir del modelo SDL se ha integrado con el sistema operativo siguiendo un modelo de integración ligera (*light*), que exige un mínimo de interacción entre ambos [STAM97]. Al compilar el código generado, la configuración a emplear se carga del fichero `user_cc.h`, que contiene la configuración de DSP/BIOS, la resolución de reloj y las variables de compilación activas.

SPICUM servicio de publicaciones

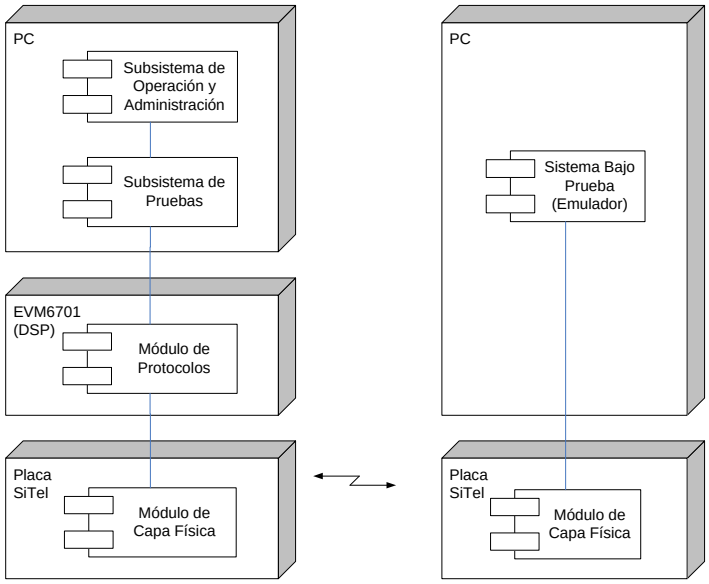


Figura 8.43: Ubicación de los componentes al emplear una plataforma de tiempo real.

El Módulo de Protocolos se ejecuta como dos tareas. La tarea principal (`Tarea_SDL`) se encarga de ejecutar el sistema SDL, incluyendo la planificación de los procesos SDL y la comunicación entre ellos, y del envío de mensajes al Subsistema de Pruebas. La segunda tarea (`Tarea_Rx`) tiene menor prioridad y se encarga de recibir mensajes del Subsistema de Pruebas. La comunicación entre ambas tareas se realiza mediante buzones con un tamaño de mensaje constante de 1025 octetos. La tarea principal cede el control a la `Tarea_Rx` cuando no tiene ninguna acción que ejecutar; la Figura 8.44 muestra gráficamente la planificación de ambas tareas. En un estudio realizado, la desviación mínima con que se cede el control a la tarea más prioritaria cuando se produce un evento ha sido menor de 40 μs.

Como Módulos Adaptador y de Gestión de los Protocolos se han empleado los descritos en el Capítulo 5. El Módulo de Gestión de los Protocolos no ha sido modificado. No obstante, el Módulo Adaptador de los Protocolos, que es el Módulo que depende más directamente de la plataforma, ha requerido leves ligera modificaciones de sus componentes:

- Gestión de Entrada/Salida: Requiere la inicialización de los canales RTDX (`RTDX_enableOutput`, `RTDX_enableInput`). La comunicación se realiza con las funciones `RTDX_write` y `RTDX_readNB`[39].
- Gestión de Tiempos: El reloj se lee con la función `clk_getltime`; se modifica el código en la función `SDL_clock`.
- Gestión de Memoria: Utiliza las funciones `malloc` y `free` de un C estándar. Por ello, sólo hace falta configurar adecuadamente los bloques de memoria disponibles.
- Gestión de Concurrencia: Se ha modificado el planificador para que la `Tarea_SDL` duerma, como máximo, hasta el siguiente vencimiento de un temporizador. Cuando se inicializa el sistema SDL, en lugar de invocar a la función `xMainLoop`, que consiste en un bucle infinito de espera de eventos del Motor SDL, se activan las interrupciones y se ejecuta la `Tarea_SDL`.

SPICUM servicio de publicaciones

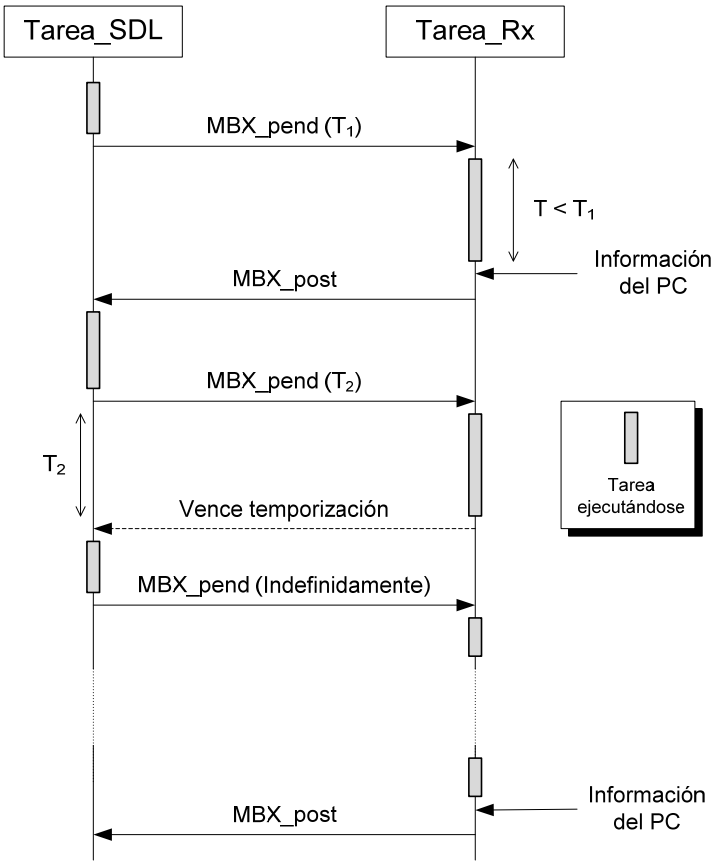


Figura 8.44: Ejemplo de secuencia de ejecución de las tareas en el DSP.

En el Subsistema de Pruebas, el Módulo Adaptador de las Pruebas ha tenido que ser adaptado para realizar la comunicación con el DSP. En concreto, se ha modificado levemente el componente de Gestión de Entrada/Salida de la siguiente forma:

- En la función `main` se inicializa la interfaz de comunicación.
- Cada PCO (bidireccional), que era un *socket*, se han implementado como dos canales RTDX (unidireccionales). Un campo en la estructura `AdPCOStruct` indica si hay datos pendientes por recibir.
- Las rutinas `GciSend` y `GciSnapshot` han sido adaptadas a la interfaz RTDX (funciones `Write`, `Read` y `GetMsgNumber`). Los paquetes intercambiados tienen un tamaño constante de 1025 octetos.

El Sistema de Pruebas obtenido ejecuta adecuadamente los Casos de Prueba del Nivel DLC de la Terminación Portátil.

8.7 Sistemas de Pruebas Comerciales

Hay pocos Sistemas de Pruebas comerciales para equipos DECT [DEWE07] (Tabla 8.38). El único suministrador que ha desarrollado una gama completa de productos para certificación, tanto de protocolo como de RF (*Radio Frequency*), ha sido Rohde & Schwarz [ROHDE]. Los productos incluidos en su catálogo han sido los siguientes:

- TS1220: Es el Sistema de Pruebas de Conformidad de Protocolos para el perfil GAP de DECT. Incluye los Juegos de Pruebas para GAP [TBR 022] y la

SPICUM servicio de publicaciones

interconexión con GSM [TBR 036] e ISDN [TBR 040]. Los Sistemas de Pruebas que se han desarrollado implementan un subconjunto de las pruebas incluidas en este equipo.

- CTS60: Es la familia de Sistemas de Pruebas de Conformidad de RF para DECT.
- PTW15: Está concebido para probar la instalación de sistemas DECT en centralitas o el bucle local.
- CMD60: Esta familia de sistemas está pensada para las pruebas de equipos en un entorno de producción.

Tabla 8.38: Sistemas de Pruebas para equipos DECT.

Suministrador	Sistema	Características
Anritsu	MT8801B	Sistema de Pruebas de RF
Advantest	R3463/5	Analizadores de espectro para pruebas de RF
Fraunhofer IIS	DTMv2	Monitor de tráfico y medidas de tasas de error (BER_T)
Hewlett Packard	HP8923B	Sistema de Pruebas de RF
Kirk Telecom	K1600	Sistema portátil para verificación de instalaciones
Rohde & Schwarz	TS1220	Sistema de Pruebas de Conformidad de Protocolos
	CTS60/65	Sistemas de Pruebas de Conformidad de RF
	CMD60	Pruebas en entorno de producción
	PTW15	Pruebas para instalaciones de centralitas o en bucle local
Tektronix	FSEM30/FSEK30	Analizadores de espectro
	SMIQ02B/03B/04B	Generadores vectoriales de señal
	SMEK2	Generador de señal
Wandel & Goltermann	WG CPM-10	Monitor de protocolos (MAC y superiores).

Otros suministradores como Anritsu, Advantest o Tektronix solamente han desarrollado sistemas para pruebas de RF. También ha habido compañías que han ofrecido equipos de medida, fuera del área de la certificación, como Kirk Telecom o Fraunhofer IIS.

8.8 Conclusiones

A lo largo de este capítulo se ha descrito la construcción de Sistemas de Pruebas de Conformidad de Protocolos para el Sistema DECT, siguiendo las fases de la Metodología de Diseño presentada en el Capítulo 4. Para realizar algunas de las actividades, nos hemos apoyado en las herramientas anexas a la Metodología, como son los módulos de adaptación a la plataforma o los generadores de interfaces y sus codificadores. Como resultado, se han obtenido Sistemas de Pruebas para los Niveles DLC y NWK de la Terminación Portátil y el Nivel DLC de la Terminación Fija. Estos sistemas han sido portados a las plataformas Unix, Linux, Windows y DSP/BIOS.

SPICUM servicio de publicaciones

Durante la exposición se ha mantenido la secuencia de actividades indicada en la Metodología para ofrecer una mejor visión del esfuerzo implicado y los resultados obtenidos de cada etapa y facilitar su comparación con la presentación, un tanto abstracta, de la Metodología en el Capítulo 4. Este desarrollo ha permitido, también, comprobar las prestaciones de algunas de las alternativas indicadas en la Metodología. Entre otros aspectos, se ha demostrado que el uso de lenguajes formales no crea problemas de velocidad durante la ejecución. La comunicación con el Módulo de Capa Física se ha realizado desde fuera del sistema SDL, pero se ha mantenido la consulta periódica del canal ascendente. Aunque esto ha sido posible en SDL por la baja velocidad de la interfaz aire, en sistemas con mayores velocidades no es una solución adecuada; este mecanismo se ha mejorado en Sistemas de Pruebas posteriores La funcionalidad incorporada a los Sistemas de Pruebas ha venido limitada fundamentalmente por las restricciones del hardware empleado, ya que no se disponía de la capacidad para modificar el *firmware* incluido en estos módulos. Esto ha motivado que no se haya desarrollado un Sistema de Pruebas para el Nivel MAC.

Respecto a los codificadores de las distintas interfaces, se realizaron pruebas para su generación en lenguaje SDL, pero su ejecución era excesivamente lenta. Por ello, se descartó esta opción, y se optó por generarlas en lenguaje C; las funciones correspondientes se invocan desde el código SDL. En la comunicación entre el Subsistema de Pruebas y los Subsistemas Inferiores se ha utilizado la codificación ASCII, mientras que en la interfaz aire se ha utilizado la codificación establecida en las normas DECT. Una de las conclusiones es que una elección correcta de las interfaces es clave para un diseño eficiente; a esto ayuda el uso de unas reglas de nombrado como las presentadas en el Capítulo 5.

Aunque en este capítulo se ha descrito cada actividad con un alto grado de detalle, ello ha sido motivado por el deseo de ofrecer una visión precisa de la aplicación de la metodología a un caso práctico real. En los siguientes capítulos se ofrecerá una visión menos detallada del trabajo realizado, presentando sólo los aspectos más destacables.

SPICUM servicio de publicaciones

Capítulo

9

"Todo proyecto de la vida real tiene sus problemas; de lo contrario, el investigador miente."

Ari, Cyteen

CAPÍTULO 9: SISTEMAS DE PRUEBAS BLUETOOTH

La experiencia alcanzada en el diseño de los Sistemas de Pruebas para DECT ha servido para validar la Metodología y las herramientas asociadas a la misma. Los resultados obtenidos han sido aplicados al diseño de un Sistema de Pruebas de protocolos para equipos Bluetooth dentro de la colaboración ([PROY02], [CONT99b], [CONT02]) con la empresa AT4 wireless [AT4W]. Esta aplicación ha permitido mejorar las herramientas y componentes utilizados con las sugerencias del grupo de diseño de la empresa. El Sistema de Pruebas resultante, BITE (*Bluetooth Qualification Tester*), se ha explotado industrialmente, siendo el Sistema de Pruebas de conformidad de protocolos para Bluetooth más vendido en todo el mundo [BITE]. La colaboración con AT4 wireless ha incluido también la validación de los Juegos de Pruebas desarrollados para varios niveles de protocolos. La tecnología Bluetooth ha sido también el ámbito en que se ha comprobado la flexibilidad de la Metodología a través el uso de entornos de desarrollo diferentes y la aplicación de la arquitectura a Sistemas de Pruebas de interoperatividad.

En este capítulo se presentan aquellos aspectos diferenciales en el diseño del Sistema de Pruebas para Bluetooth respecto al diseño descrito en el capítulo anterior. Además, se muestra que la Metodología es independiente de entornos de desarrollo particulares. Para ello, se han construido Sistemas de Pruebas para los niveles LM y SPP [SORE03a] en los que el Subsistema de Pruebas se ha generado con una herramienta distinta (*TTCN Toolbox*) [DANET]. La integración del Subsistema Inferior ha requerido la adaptación de las interfaces ofrecidas por la nueva herramienta. Por otro lado, las pruebas de conformidad se han definido tan sólo para los niveles inferiores de protocolo, mientras que para los perfiles, niveles más próximos al usuario, se ha considerado más adecuado la definición de pruebas de interoperatividad. Por este motivo, se ha considerado interesante, dentro de la colaboración con AT4 wireless, el diseño de un Sistema de Pruebas de interoperatividad, en concreto para el perfil *Headset* ([MORI03], [MORI02a]). Aunque la arquitectura de los Sistemas de Pruebas ha sido concebida para el diseño de Sistemas de Pruebas de conformidad, se demuestra aquí que también es aplicable al área de las pruebas de interoperatividad. La arquitectura es independiente del lenguaje utilizado en cada uno de los componentes; como se verá, se ha utilizado como Módulo

SPICUM servicio de publicaciones

de Protocolos del Sistema de Pruebas de interoperatividad un software de libre distribución.

A lo largo de este capítulo se describen los aspectos más importantes de la construcción de cada uno de los Sistemas de Pruebas. En primer lugar, se presenta un resumen de la tecnología Bluetooth. A continuación, se explican los puntos novedosos en el diseño del Sistema de Pruebas comercial. Las secciones posteriores describen la construcción de los Sistemas de Pruebas de conformidad para los niveles LM y L2CAP y del Sistema de Pruebas de interoperatividad para el perfil *Headset*. Finalmente, se resumen brevemente las características de los Sistemas de Pruebas comercializados.

9.1 Descripción del Sistema Bluetooth

Bluetooth ([BT CORE], [BT PROF], [BT OVER]) es una tecnología de comunicaciones inalámbricas de corto alcance, de 10 a 100 m, en la banda libre ISM (*Industrial, Scientific & Medical*) de 2,4000 a 2,4835 GHz, surgida en la segunda mitad de los años noventa bajo el amparo de varios de los fabricantes más importantes del mundo[1]. El objetivo buscado era una tecnología de bajo coste y baja potencia en la banda libre que permitiera a un dispositivo conectarse a los servicios disponibles en el área de su localización.

La transmisión se realiza a una velocidad, en la interfaz aire, de 1 Mbit/s en TDD, con ranuras de 625 μs asignadas alternativamente a transmisión y recepción; los paquetes pueden ocupar desde una hasta cinco ranuras. Para evitar problemas de interferencia y desvanecimiento utiliza el mecanismo de saltos de frecuencia (1600 saltos/segundo), empleando 23 ó 79 frecuencias diferentes según el país. Los dispositivos Bluetooth adaptan la potencia transmitida según la distancia a la que se encuentre la unidad receptora mediante un proceso de realimentación por el cual el receptor indica al transmisor que aumente o disminuya su potencia.

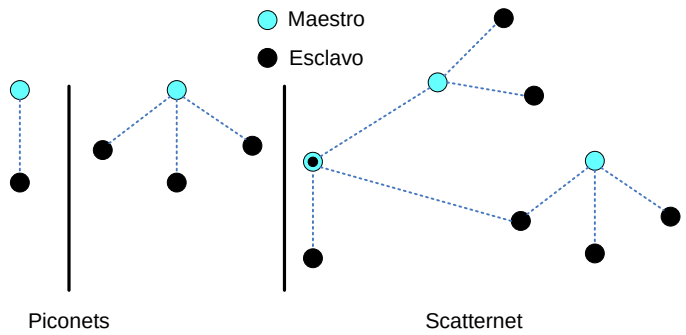


Figura 9.1: Ejemplos de redes con dispositivos Bluetooth.

Un equipo Bluetooth en comunicación con otro forma parte de una *piconet* (ver Figura 9.1). Se trata de una red de hasta 8 equipos activos simultáneamente, donde uno de ellos hace de maestro de la red, mientras que los demás actúan de esclavos. El actuar en un rol o en otro influye en los procedimientos de inicio de una comunicación; además, los

[1] El organismo que promueve este tecnología es el SIG (*Bluetooth Special Interest Group*), creado en 1998 por los promotores originales de la tecnología: Ericsson, Nokia, Toshiba, Intel e IBM.

SPICUM servicio de publicaciones

esclavos deben sincronizarse con el reloj del dispositivo maestro. Dos o más *piconets* conectadas forman una *scatternet*.

Inicialmente, los dispositivos Bluetooth se encuentran en el modo *standby*, estado en el que se consume la menor potencia. En este estado, el dispositivo busca mensajes cada 1,28 segundos. Para descubrir los dispositivos presentes (sus direcciones) en la zona de cobertura, se realiza un proceso de búsqueda (*inquiry*). El establecimiento de la conexión se realiza mediante el procedimiento de *paging*. La Figura 9.2 muestra los estados que atraviesa un dispositivo Bluetooth durante el proceso de conexión. Una vez alcanzado el estado activo, un dispositivo esclavo puede pasar a uno de los siguientes estados de bajo consumo (en orden descendente de consumo):

- *Sniff*: El dispositivo esclavo sólo escucha en instantes determinados; el maestro sólo puede transmitir datos al esclavo en dichos instantes.
- *Hold*: Se suspende toda comunicación asíncrona y el dispositivo esclavo sólo escucha cada cierto tiempo para saber si tiene que volver al estado activo.
- *Park*: El dispositivo esclavo continúa sincronizado a la *piconet*, pero no recibe ni transmite datos. Puede haber en este modo hasta 255 dispositivos esclavos en la misma celda.

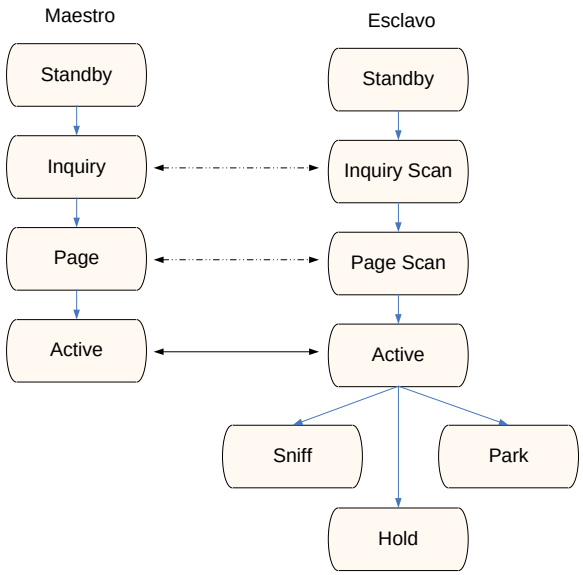


Figura 9.2: Proceso de conexión y estados en que puede encontrarse un dispositivo Bluetooth.

Una vez realizada la conexión entre dos dispositivos, se pueden establecer dos tipos de enlaces:

- Síncronos (SCO – *Synchronous Connection Oriented*): enlaces punto a punto, simétricos, pensados para transmisión de voz.
- Asíncronos (ACL – *Asynchronous ConnectionLess*): enlaces punto a multipunto en los que no hay reserva de ranuras temporales. Está enfocado a la transmisión de datos.

Existen diversas versiones de la tecnología Bluetooth (v1.0, v1.0B, v1.1, v1.2, v2.0+EDR y v2.1). Las versiones 1.1 y 1.2 han sido adoptadas por IEEE como estándar

SPICUM servicio de publicaciones

IEEE 802.15-1 ([IEEE 802.15-1]). Los dispositivos versión 1.x pueden soportar un canal asíncrono y hasta tres canales síncronos de voz (64 kbit/s en cada sentido). La velocidad máxima que un usuario puede conseguir son 723,2 kbit/s (en un solo sentido), con 57,6 kbit/s en el sentido opuesto. La versión 2.0 fue publicada en 2004 y su principal mejora es el aumento de la velocidad de transferencia (EDR – *Enhanced Data Rate*) hasta 3 Mbit/s, que se convierten en una tasa neta de unos 2,1 Mbit/s en la práctica. Además, introduce mecanismos para reducir el consumo de potencia.

Entre los competidores principales de la tecnología Bluetooth se encuentran tanto DECT como el estándar IEEE 802.11; otras tecnologías competidoras, aunque con menor influencia, han sido los infrarrojos (IrDA) y HomeRF. En la Tabla 9.1 se ofrece una comparativa de las características principales de cada una de estas tecnologías.

Tabla 9.1: Comparación entre tecnologías de comunicaciones inalámbricas [ITCO02].

Tecnología	Velocidad (Mbps)	Banda (GHz)	Alcance (metros)
IrDA	16	-	3-5
DECT	1	1,88	300 – 25000
Bluetooth	1	2,4	10 – 100
	3	2,4	10 – 100
HomeRF	1,6	2,4	50
	10	2,4	50
IEEE 802.11	2	2,4	100
	11	2,4	100
	24 – 54	5	25 – 600

9.1.1 Arquitectura

La arquitectura Bluetooth se divide en núcleo y perfiles (Figura 9.3). El núcleo está formado por las capas básicas del sistema, como son:

- Banda Base (BB – *BaseBand*) [BT BB]: Adapta la señal para su transmisión por la interfaz aire. Se encarga de manejar los canales y enlaces físicos, corregir errores, elegir el patrón de saltos y de asegurar (autentificación y cifrado) la comunicación.
- Gestor del Enlace (LM – *Link Manager*) [BT LM]: Es responsable del establecimiento, configuración y supervisión de los enlaces, la gestión de la seguridad (autentificación y cifrado), la monitorización de la calidad de servicio y el control del estado del Nivel Banda Base. Asimismo, se encarga de controlar los procesos de búsqueda y conexión, de gestionar la picocelda (modos de ahorro de energía, cambios de roles, ...) y de controlar el manejo de los paquetes multiranura. Se comunica con otras entidades LM a través del protocolo LMP (*Link Manager Protocol*), con paquetes de mayor prioridad que los paquetes L2CAP y que ocupan una única ranura.
- Control Lógico y Adaptación del Enlace (L2CAP – *Logical Link Control and Adaptation Protocol*) [BT L2CAP]: Se encarga de multiplexar los protocolos de nivel superior, segmentar y reensamblar los paquetes, gestionar la calidad de servicio y gestionar grupos de dispositivos. Únicamente maneja paquetes ACL,

SPICUM servicio de publicaciones

aunque gestiona tanto enlaces orientados a conexión como no orientados a conexión.

- Servicio de Descubrimiento (SDP – *Service Discovery Protocol*) [BT SDP]: Este servicio se utiliza para determinar qué equipos de la red ofrecen un servicio determinado, ya que los equipos aparecen y desaparecen dinámicamente.
- Comunicación por Frecuencia Radio (RFCOMM – *Radio Frequency COMMunication*) [BT RFCOMM]: Es un protocolo de transporte que permite emular puertos serie sobre el protocolo L2CAP.
- Especificación del protocolo de Control de Telefonía (TCS – *Telephony Control protocol Specification*) [BT TCS]: Define el proceso de señalización para el establecimiento de llamadas, tanto de voz como de datos, entre dos dispositivos Bluetooth.

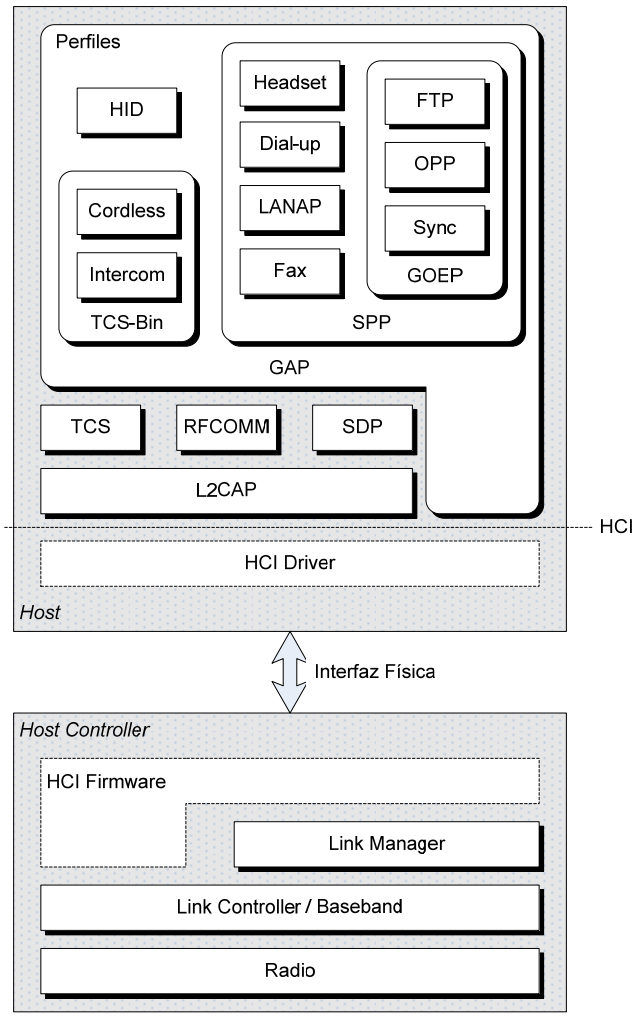


Figura 9.3: Arquitectura Bluetooth.

El núcleo a su vez, se divide en dos partes. Por un lado, se agrupan las capas banda base y de gestión del enlace, formando lo que se denomina *Host Controller*; por otro, se encuentran las capas de control lógico y adaptación del enlace, el servicio de descubrimiento y el emulador de puertos serie. Esta subdivisión está directamente

SPICUM servicio de publicaciones

relacionada con la implementación física de los dispositivos. Los dispositivos Bluetooth comerciales, no integrados en otros equipos como cámaras, PDAs, etc., incluyen las dos capas inferiores; el resto de las capas se implementan dentro del equipo (*host*) al que se conecte el dispositivo Bluetooth.

Entre ambos grupos se define la interfaz HCI (*Host Controller Interface*) [BT HCI]. Esta interfaz ofrece un conjunto de comandos que permiten acceder a la funcionalidad de las capas inferiores ocultando la interfaz física (USB, UART o RS232)[2]. Funcionalmente se encuentra dividida en tres secciones, ubicadas en diferentes partes del sistema: *HCI firmware*, capa de transporte (*Host Controller Transport Layer*) y *HCI Driver*. El *HCI firmware* se encuentra en el *Host Controller*, implementa los comandos HCI sobre el *hardware* Bluetooth mediante comandos *Baseband* y *Link Manager*, y accede a los registros de estado, de control y de eventos. El *HCI Driver* es un *software* que se encuentra en el *Host*, cuya función es recibir las notificaciones de eventos. *Host Controller Transport Layer* (HCTL) es la interfaz física a través de la cual se comunican el *Host Driver* y el *Host Firmware*. La interfaz HCI no siempre hace falta, ya que la subdivisión del núcleo es opcional.

Por encima del núcleo se sitúan los perfiles. Bluetooth define distintos perfiles, que cada equipo implementará dependiendo de su funcionalidad. Estos perfiles pueden, a su vez, hacer uso de otros perfiles para implementar los servicios que definan. En [METT99] se comentan posibles modelos de uso. Algunos perfiles que pueden emplearse son:

- Perfil de Acceso Genérico (GAP – *Generic Access Profile*) [BT GAP]: Ofrece procedimientos genéricos para descubrir dispositivos Bluetooth y gestionar las conexiones. Es la base del resto de perfiles.
- Perfil de Puerto Serie (SPP – *Serial Port Profile*) [BT SPP]: Permite establecer conexiones a través de una emulación de puerto serie.
- Perfil de Auriculares (HP – *Headset Profile*) [BT HEAD]: Define la funcionalidad que debe implementar un equipo Bluetooth que funcione como auricular, o que le dé servicio.
- Perfil de Acceso LAN (LANAP – *Local Area Network Access Profile*) [BT LANAP]: Define los mecanismos para acceder a redes LAN mediante PPP (*Point-to-Point Protocol*) sobre dispositivos Bluetooth.
- Perfil de Intercambio de Objetos Genéricos (GOEP – *Generic Object Exchange Profile*) [BT GOEP]: Especifica los procedimientos genéricos para el intercambio de objetos entre dos equipos Bluetooth. Este perfil implementa el protocolo OBEX[3] (*OBject eXchange Protocol*) de acuerdo a la definición proporcionada por la asociación de datos infrarrojos (IrDA) [IRDA03].
- Perfil de Envío de Objetos (OPP – *Object Push Profile*) [BT OPP]: Define los servicios de envío de objetos, obtención e intercambio de tarjetas de visita.
- Perfil de Transferencia de Ficheros (FTP – *File Transfer Profile*) [BT FTP]: Define los mecanismos para navegar por carpetas, transferir carpetas y ficheros y manipular objetos del servidor (borrar, crear, etc.).

[2] USB – *Universal Serial Bus*; UART – *Universal Asynchronous Receiver-Transmitter*.

[3] En las figuras se ha preferido emplear el nombre del protocolo en lugar el nombre del perfil ya que es la denominación utilizada generalmente.

SPICUM servicio de publicaciones

- Perfil de Dispositivos de Interfaz Humana (HID – *Human Interface Devices*) [BT HID]: permite la comunicación entre dispositivos periféricos (ratones, teclados, *joystick*) y un equipo central. También permite comunicarse con dispositivos (termómetros, lectores de códigos de barras) que no requieren interacción humana.

9.2 Sistema de Pruebas de Conformidad

Se ha colaborado con AT4 wireless en el diseño de un Sistema de Pruebas de Conformidad de protocolos para Bluetooth, que se ha comercializado bajo el nombre de BITE [BITE]. Este Sistema de Pruebas se ha construido en base a los principios y herramientas expuestos en capítulos anteriores. Se ha participado en el diseño del Módulo de Protocolos y, tanto en el Subsistema de Pruebas como en el Subsistema Inferior, se han utilizado los Módulos Adaptador y de Gestión desarrollados como herramientas de la Metodología. La comunicación entre el Subsistema de Pruebas y el Subsistema Inferior emplea codificación ASCII, habiéndose utilizado la herramienta *GenInt* para construir el elemento Codec $\text{Local}_{\text{TTCN}}$ del Subsistema Inferior.

Los Métodos de Pruebas utilizados para verificar cada uno de los niveles y protocolos de Bluetooth emplean tres Subsistemas Inferiores diferentes. La Figura 9.4 muestra varios de estos Métodos de Pruebas; en el Apéndice C se incluyen todos los Métodos de Pruebas. Para certificar los niveles *Baseband*, LM y el perfil GAP el Subsistema Inferior contiene solamente el *Baseband* y la radio; para certificar el protocolo L2CAP añade el nivel LM; para certificar el protocolo SDP y el perfil SPP se añade el protocolo L2CAP. Por ello, sólo en el último caso existe un Módulo de Protocolos como tal en la arquitectura del Sistema de Pruebas. Dado que los niveles BB y LM se realizan dentro del *Host Controller*, el Módulo de Protocolos está constituido por el protocolo L2CAP y la interfaz HCI.

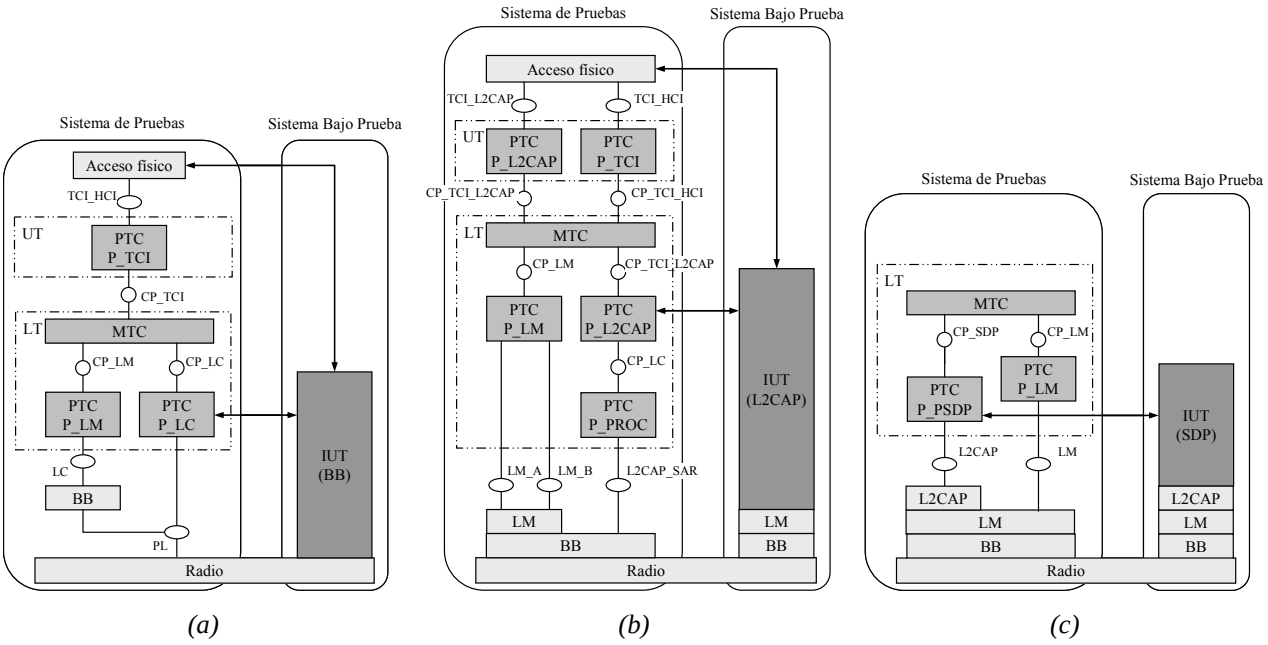


Figura 9.4: Métodos de Pruebas para los Niveles (a) Banda Base, (b) L2CAP y (c) SDP.

SPICUM servicio de publicaciones

Se describen a continuación únicamente aquellos aspectos que representan una diferencia respecto al diseño de los Sistemas de Pruebas de DECT.

9.2.1 Módulo de Protocolos

La estructura del Módulo de Protocolos es la mostrada en la Figura 9.5. Consta de 3 procesos que implementan el nivel L2CAP (`L2CAP`), la interfaz HCI (`HCI`) y un gestor (`GESTOR`) que inicia y puede terminar la operación de ambos.

El diseño del nivel L2CAP (Figura 9.6-a) se ha realizado [SANC00b] separando la funcionalidad propia del nivel (`L2CAP_Layer`), donde se crea un proceso por cada canal establecido, de la codificación necesaria para comunicarse con el nivel HCI (`L2CAP_Driver`), que transforma las primitivas L2CAP a comandos HCI y viceversa. La implementación se ha verificado con las pruebas del SIG.

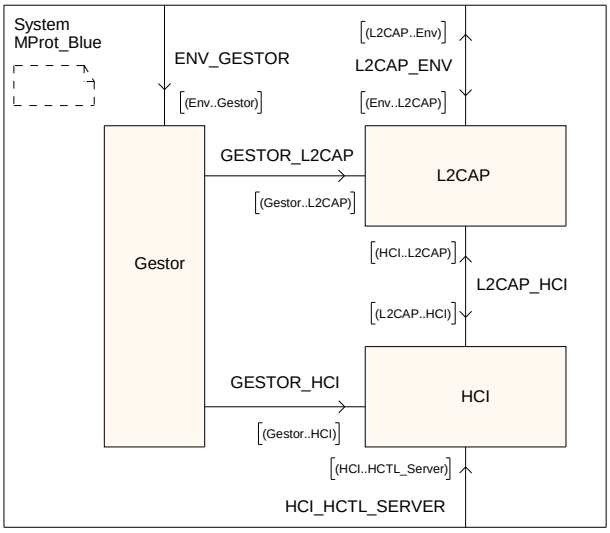


Figura 9.5: Estructura del Módulo de Protocolos para Bluetooth.

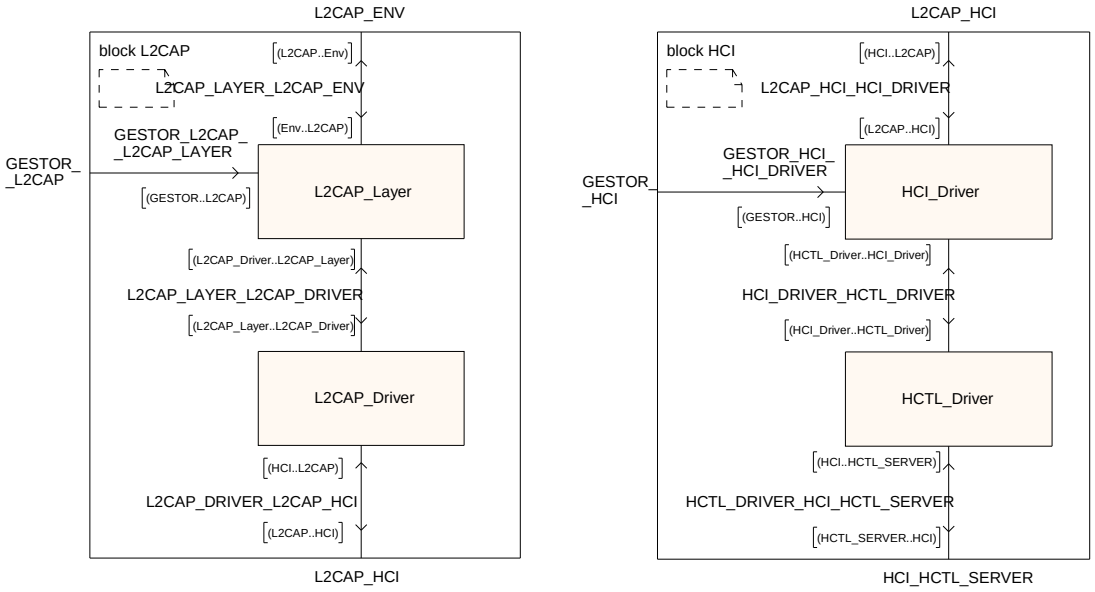
El bloque HCI (Figura 9.6-b) sigue internamente la división propuesta en la norma [GARC00], estructurándose en un bloque que gestiona los paquetes HCI (`HCI_Driver`) y otro bloque que realiza la comunicación por el medio físico (`HCTL_Driver`), en este caso una UART. Ambos bloques están formados por dos procesos, actuando uno de transmisor y el otro de receptor. La comunicación con el *Host Controller* se realiza a través de la aplicación externa *HCTL Server*. La comunicación entre el sistema SDL y esta aplicación se realiza, como en el Sistema de Pruebas de DECT, mediante llamadas a funciones C desde los procesos del bloque `HCTL_Driver`; la cola de recepción se comprueba cada 100 ms. La codificación y descodificación de los paquetes se realiza mediante procedimientos construidos en C.

La aplicación *HCTL Server* implementa la funcionalidad de la capa de transporte *Host Controller Transport Layer* (la interfaz física). La comunicación con el *Host Controller* se realiza a través de un puerto hardware USB o RS-232; la comunicación con los niveles superiores a la interfaz HCI se realiza mediante un *socket*, siendo esta aplicación el servidor. Un hilo controla la llegada de datos por el puerto hardware, mientras que otro lo hace por el *socket*. Esta aplicación ha sido posteriormente ampliada por AT4 wireless.

SPICUM servicio de publicaciones

La integración entre los niveles L2CAP y HCI se ha verificado, en primer lugar, mediante la emulación de los niveles por debajo de HCI. Posteriormente, se ha integrado el Módulo de Protocolos con un Módulo de Capa Física de Ericsson (EBDK – *Ericsson Bluetooth Development Kit*) [EBDK01] y se ha comunicado con otros equipos.

Este Módulo de Protocolos ha sido completado más tarde con implementaciones en SDL de SDP, RFCOMM, SPP y HID y en C de GAP y OBEX ([VIGO04], [ROME03], [TERN03], [LARA04]). A partir del sistema completo se ha generado una DLL que ofrece una interfaz en C++ a las aplicaciones ([FERN04], [GIMI05], [ESPI05], [SERR06]). También se ha construido un controlador USB [CARD05] para usar dispositivos comerciales con la pila Bluetooth desarrollada, pero no se ha integrado en el Sistema de Pruebas.



(a) *(b)*

Figura 9.6: Estructura de los bloques (a) L2CAP y (b) HCI.

9.2.2 Evolución

Este Sistema de Pruebas incluye mejoras en algunos de los componentes utilizados respecto a los empleados en los Sistemas de Pruebas para DECT.

En el Subsistema de Pruebas, se ha incluido el componente de Gestión de Concurrencia, que no era utilizado previamente. Este componente crea un hilo (`AdNuevoThread`) para cada Componente de Prueba (MTC o PTC) y permite identificar el componente a que corresponde mediante el identificador del hilo (`GetCompNameByThreadId`). Además, ofrece un servicio de tareas protegidas con semáforos; las funciones que ofrece son de inicialización (`taskinit`), creación (`taskcreate`) y destrucción (`taskdelete`) de tareas.

Los Juegos de Pruebas para Bluetooth hacen uso de un reducido número de tipos de datos definidos en ASN.1, que se emplean tanto para definir campos de PDUs como Parámetros de Pruebas. En el primer caso afectan al codificador entre el Subsistema de Pruebas y el Subsistema Inferior (componentes Codec $Local_{TTCN}$). El componente del Subsistema de Pruebas no ha sido necesario modificarlo, ya que la API ofrecida por la

SPICUM servicio de publicaciones

herramienta de generación de código maneja de forma equivalente los tipos TTCN y los tipos ASN.1. El componente del Subsistema Inferior se ha generado con la herramienta *GenInt* y se ha modificado posteriormente para incluir la información necesaria para poder manejar los tipos ASN.1. En el caso de los Parámetros de Pruebas, la función de lectura de los mismos no requiere modificaciones, pues la sintaxis de escritura de los valores es la misma que para los tipos estructurados de TTCN.

Se ha retocado el componente de Gestión de Entrada/Salida del Subsistema Inferior, incluyendo la posibilidad de enviar al Subsistema de Operación y Administración trazas informativas de la ejecución del Módulo de Protocolos. La función `xSignalLog` traza los eventos fuera del ámbito de un proceso, mientras que la función `xProcessLog` traza los eventos dentro de su ámbito. También se ha mejorado la organización del código generado automáticamente.

9.3 Uso de una Herramienta Comercial Alternativa

Además de la construcción de un Sistema de Pruebas comercial descrita en la Sección anterior, se ha evaluado el uso de herramientas alternativas para la generación del Subsistema de Pruebas. Se ha mostrado así que la arquitectura propuesta de un Sistema de Pruebas no depende de herramientas específicas. Se ha evaluado la herramienta *TTCN Toolbox* [DANET] como sustituta de la herramienta *Tau Suite*, buscando reducir los costes de desarrollo ya que las licencias de la primera eran bastante más baratas.

Se han realizado Sistemas de Pruebas para los niveles SPP y LM ([MORI02a], [SORE03a]). Al emplear una herramienta diferente, ha habido que adaptar la interfaz del Subsistema de Pruebas generada por la nueva herramienta para realizar la comunicación con el Subsistema Inferior. Además, como Módulos Adaptador y de Gestión de las Pruebas se han empleado los proporcionados por la herramienta. Igualmente, se ha utilizado una aplicación de la herramienta para controlar la ejecución de los Casos de Prueba.

Los Subsistemas de Pruebas se han construido sobre la plataforma Windows 2000 y se ha utilizado el compilador Cygwin [CYGWIN]. Como Módulo de Capa Física se han empleado un dispositivo de CSR (*Cambridge Silicon Radio*) [CSR02], el kit de desarrollo EBDK (*Ericsson Bluetooth Development Kit*) [EBDK01] y un prototipo del Módulo de Capa Física incluido en el Sistema de Pruebas comercial; como IUT se ha empleado un dispositivo comercial.

A continuación se describen los Sistemas de Pruebas construidos.

9.3.1 Perfil de Puerto Serie (SPP)

El Perfil de Puerto Serie (SPP – *Serial Port Profile*) [BT SPP] define los requisitos necesarios para establecer un puerto serie emulado entre dos dispositivos Bluetooth utilizando RFCOMM. RFCOMM [BT RFCOMM] es un protocolo de transporte, basado en el estándar de ETSI [TS 07.10], que permite emular puertos serie sobre el protocolo L2CAP. Cada puerto serie abierto corresponde a una conexión de enlace (DLC – *Data Link Connection*), identificada por el DLCI (*Data Link Connection Identifier*)[4]. Sobre este protocolo trabaja el perfil SPP, que define dos roles: DevA, iniciador en el establecimiento de la conexión, y DevB, el destinatario de la conexión.

[4] Es un identificador de 6 bits, empleando el rango [2..61], lo que hace que pueda haber hasta 60 conexiones simultáneas entre dos dispositivos.

SPICUM servicio de publicaciones

El Juego de Pruebas incluye 6 Casos de Prueba aplicables al perfil SPP y 23 Casos de Prueba aplicables al protocolo RFCOMM [BTEST SPP]. Se utiliza el Método de Pruebas remoto. La Figura 9.7 muestra la estructura interna de los Componentes de Pruebas y la distribución de la funcionalidad entre el Módulo de Protocolos y el Módulo de Capa Física. Como Módulo de Protocolos se ha empleado el incluido en el Sistema de Pruebas comercial (ver Sección 9.2.1); como Módulo de Capa Física se ha utilizado el *Host Controller* de CSR.

Para las Pruebas de Sistema se ha utilizado como IUT la pila Axis OpenBT sobre plataforma Windows (Sección 9.4) y el *Host Controller* de CSR. No se han ejecutado los Casos de Prueba que requieren situar a la IUT en un modo de bajo consumo (`TP/APP/BV/03-C`, `TP/APP/BV/04-C`, `TP/APP/BV/05-C`) porque no es posible configurar externamente estos modos en la versión del módulo utilizado.

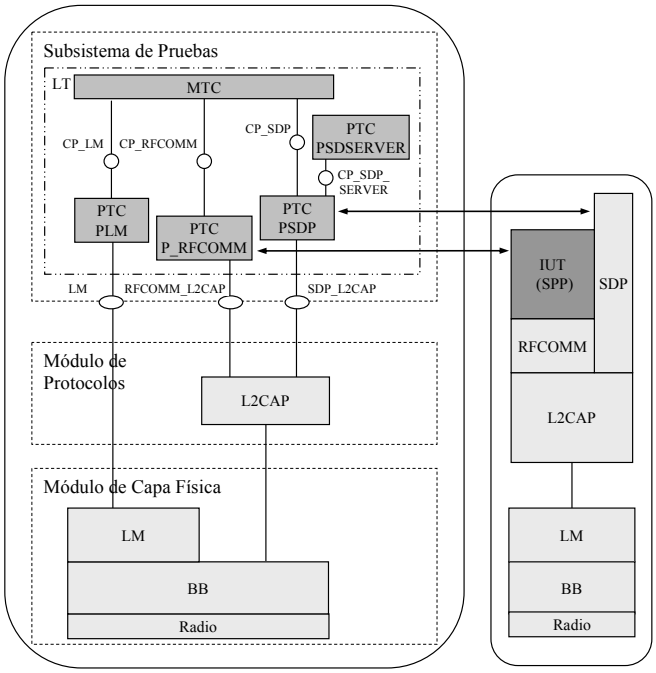


Figura 9.7: Asignación de funcionalidad a cada Subsistema del Sistema de Pruebas para SPP.

9.3.1.1 Subsistema de Pruebas

La herramienta *TTCN Toolbox* genera una carpeta para cada Caso de Prueba donde incluye un fichero .c para cada Componente de Prueba. El componente Codec $Local_{TTCN}$ se construye como uno o más archivos .c (TYPE_FUNCTIONS<n>.c) donde aparecen una función de codificación (`enc_<tipo>`) y otra de descodificación (`check_<tipo>`) para cada ASP, PDU, CM (*Coordination Message*) o tipo de dato. La Figura 9.8 muestra el uso de la función de codificación de la primitiva `L2CA_DATA_REQ` [SORE02]. Este código generado por la herramienta emplea, en la interfaz con el Subsistema Inferior, una sintaxis de transferencia binaria basada en las reglas de codificación BER. Sin embargo, es una sintaxis ambigua, ya que no es capaz de identificar los campos omitidos al codificar un valor.

SPICUM servicio de publicaciones

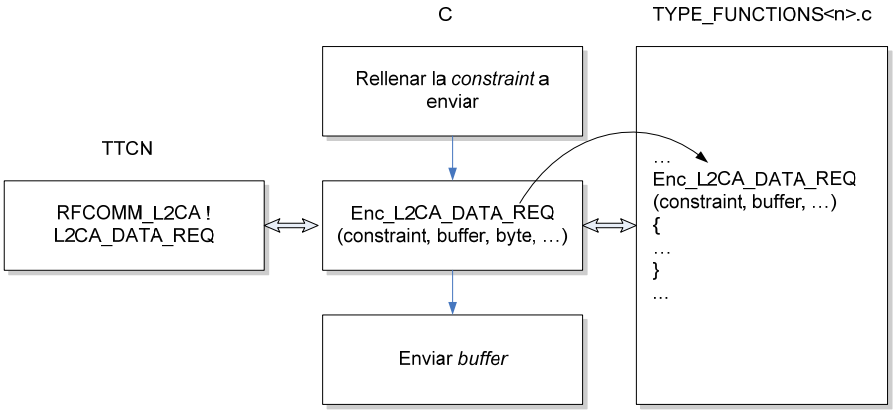


Figura 9.8: Invocación de la función de codificación antes de enviar una primitiva.

La sintaxis de transferencia que proporciona la herramienta es diferente de la empleada por el Subsistema Inferior. Por este motivo, se han generado externamente las funciones de codificación y descodificación[5] creadas por *TTCN Toolbox* [SORE02]. Para ello, se ha modificado la herramienta *GenCod* para que genere estas funciones, manteniendo la cabecera, a partir de los tipos de datos extraídos por *GenDef* del Juego de Pruebas. Los tipos ASN.1 se han incluido manualmente en las salidas de *GenDef*. Las trazas han sido adaptadas a la nueva sintaxis de transferencia.

9.3.2 Gestor del Enlace (LM)

El Nivel de Gestor del Enlace (LM – *Link Manager*) [BT LM] se encarga fundamentalmente del establecimiento y control del enlace, de la gestión de potencia (dinámica y modos de bajo consumo) y de la seguridad de las comunicaciones. Para ello hace uso del protocolo LMP (*Link Manager Protocol*), mediante el cual se comunica con el Nivel LM de otro dispositivo. Las PDUs del protocolo se transmiten en paquetes DM1, excepto en el caso de existan conexiones SCO, donde podrían usarse paquetes DV. Entre otros aspectos, el protocolo LMP permite sincronizar los relojes del maestro y el esclavo. Los mensajes de este nivel siempre tienen mayor prioridad que los datos de usuario. Una implementación de este Nivel se describe en [LLAB00].

El Juego de Pruebas incluye 104 Casos de Prueba organizados en cinco grupos según la funcionalidad que verifican: autentificación (16), cifrado (9), peticiones de información (10), gestión del enlace (68) y modo de prueba (1) [BTEST LM]. Se utiliza el Método de Pruebas local. Los Casos de Prueba implementados son todos los pertenecientes a los grupos de información y modo de pruebas, y parte de los incluidos en los grupos de autentificación y gestión del enlace.

La Figura 9.9 muestra la estructura interna de los Componentes de Pruebas y la distribución de la funcionalidad entre el Módulo de Protocolos y el Módulo de Capa Física. Como Módulo de Capa Física se ha utilizado una versión preliminar del incluido en el Sistema de Pruebas comercial. Como se observa, el Módulo de Protocolos no incluye ningún nivel; los PCOs del Juego de Pruebas acceden directamente al Nivel Banda Base, que forma parte del *Host Controller*, el cual pertenece al Módulo de Capa Física de la arquitectura del Sistema de Pruebas. Por ello, como se explica en la Sección

[5] Se ha utilizado la sintaxis de transferencia ASCII (Capítulo 5, Sección 5.5.1.2.1.1).

SPICUM servicio de publicaciones

0, el Subsistema de Pruebas se ha conectado directamente con el Módulo de Capa Física.

El Subsistema de Pruebas posee tres PCOs, dos de los cuales se comunican con el Subsistema Inferior (P_LC, P_LM), mientras que el tercero se comunica con la frontera superior de la Implementación Bajo Prueba (TCI_HCI). La interfaz ofrecida en cada uno de los PCOs es una interfaz HCI. Esta interfaz es la indicada para el PCO TCI_HCI, mientras que para los otros PCO se ha elegido así pues la comunicación con el Módulo de Capa Física se realiza a través de la aplicación *HCTL Server* (Sección 9.2.1).

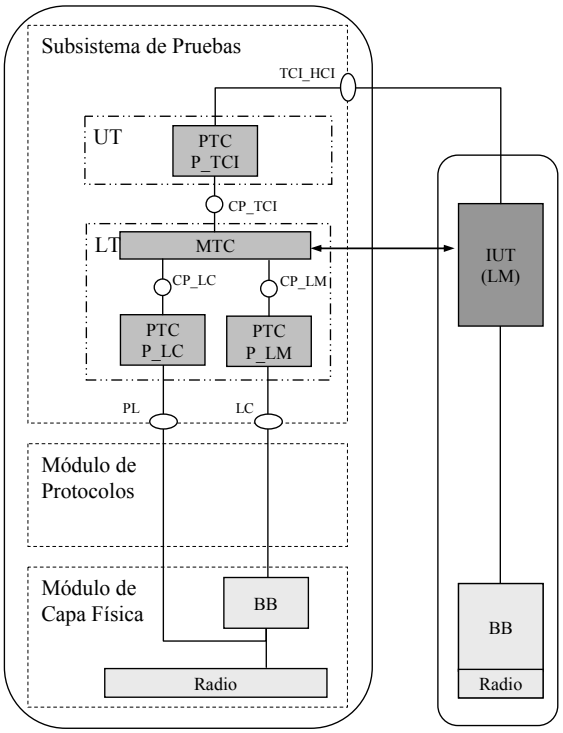


Figura 9.9: Asignación de funcionalidad a cada Subsistema del Sistema de Pruebas para LM.

9.3.2.1 Subsistema de Pruebas

Como en el caso del perfil SPP, la sintaxis de transferencia que ofrece el código generado por *TTCN Toolbox* no es el esperado por el Subsistema Inferior. Se plantea la disyuntiva de si modificar esta sintaxis para comunicar el Subsistema de Pruebas con el Módulo de Protocolos o si comunicarlo directamente con el Módulo de Capa Física. Al fin y al cabo, el Módulo de Protocolos se encuentra vacío, pues no implementa ningún nivel de protocolo, y hay que diseñar el componente Codec $Local_{PHY}$ para comunicarlo con el Módulo de Capa Física.

Se ha optado por eliminar el Módulo de Protocolos y modificar la sintaxis de transferencia para comunicar el Subsistema de Pruebas directamente con el Módulo de Capa Física. La interfaz esperada por el Módulo de Capa Física se encuentra descrita en [SANC02].

Las modificaciones se han incorporado a las librerías de *TTCN Toolbox* ya que implica una menor complejidad que crear funciones propias de codificación (`enc_<tipo>`) y

SPICUM servicio de publicaciones

descodificación (`check_<tipo>`). Una vez recompiladas, los cambios se aplican automáticamente a cualquier Juego de Pruebas para el que se genere código posteriormente. Para aquellos aspectos que no se han podido resolver modificando estas librerías, se han creado nuevas funciones que se enlazan con el código generado; por lo general, estas nuevas funciones sustituyen a otras generadas. Finalmente, allí donde la complejidad de las modificaciones necesarias es la misma que la de su anulación en el código generado, se ha optado por realizar directamente las modificaciones en el código generado[6]. La Figura 9.10 muestra el flujo de decisión acerca de qué tipo de modificación realizar.

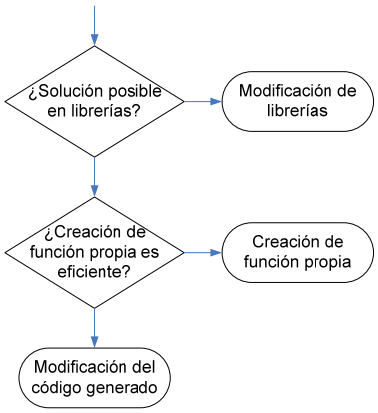


Figura 9.10: Criterios seguidos para la modificación del código.

Las modificaciones realizadas a las librerías son las siguientes:

- Emplear el formato *Little Endian* en las funciones de codificación y descodificación de cadenas de octetos y bits. *TTCN Toolbox* emplea el formato *Big Endian*. Debe mantenerse el formato *Big Endian* para las cadenas de texto.
- El formato de codificación empleado por una PDU es diferente del utilizado en un ASP. *TTCN Toolbox* emplea el mismo formato para ambas.
- Usar codificación directa para todos los tipos de datos, incluyendo los de ASN.1. *TTCN Toolbox*, por defecto, codifica los tipos ASN.1 con reglas BER.
- Preceder los mensajes con campos no presentes en el ATS requeridos por los otros elementos del Sistema de Pruebas.

La codificación en formato *Little Endian* se ha incorporado modificando las funciones básicas de codificación de cadenas de octetos y bits. Por cada función previa de la librería `libcb.a` se han creado dos funciones, se ha cambiado el nombre a la función original añadiéndole un prefijo `_BE_` y se ha creado una función nueva, con el nombre original para que sea la función empleada por defecto, que codifica en formato *Little Endian*. La figura 6.3 muestra, a modo de ejemplo, el resultado de codificar la cadena de octetos `0xAABBCCDD` en formato *Big Endian* y en *Little Endian*. La Tabla 9.2 muestra las funciones modificadas y los cambios realizados en cada una de ellas. Una vez realizadas las modificaciones se recompila la librería `libcb.a`.

[6] Por ejemplo, la selección de la regla de codificación adecuada de los tipos de datos ASN.1.

SPICUM servicio de publicaciones

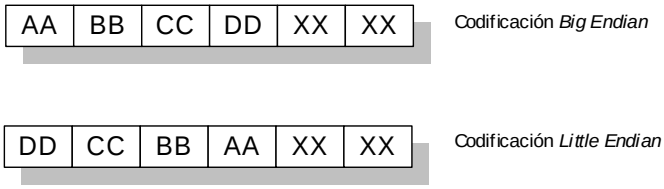


Figura 9.11: Codificación Big Endian y Little Endian.

La diferencia entre la codificación de ASPs y PDUs es que los parámetros de las ASPs deben usar un número entero de octetos. Por ello, se han creado nuevas funciones, con prefijo `ASP_`, para codificar los parámetros de una ASP que sean cadenas de bits. Estas funciones se muestran en la Tabla 9.2. Para codificar los tipos ASN.1 en forma directa, y no en BER como se hace por defecto, se ha modificado la sentencia que indica el tipo de codificación o descodificación a realizar al inicio de cada función; la regla a utilizar es la `encode_DIRECT`.

Tabla 9.2: Funciones modificadas para incorporar el formato Little Endian.

Original	Creadas	Cambio
`enc_OctetString`	`enc_OctetString`	Añadir al final una llamada a la función `L_E` para invertir el orden de los octetos.
	`enc_BE_OctetString`	Función original.
`check_OctetString`	`check_OctetString`	Cambiar el índice del bucle que procesa el valor recibido.
	`check_BE_OctetString`	Función original.
`enc_BitString`	`enc_BitString`	Sustituir la regla de codificación por defecto por la regla `encode_ISUP`[7].
	`enc_BE_BitString`	Función original.
	`ASP_enc_BitString`	Rellena con ceros hasta completar un número entero de octetos.
`check_BitString`	`check_BitString`	Sustituir la regla de codificación por defecto por la regla `encode_ISUP`[7].
	`check_BE_BitString`	Función original.
	`ASP_check_BitString`	Descodifica el valor y elimina los bits de relleno.

Los mensajes que circulan por los PCOs LC y PL, los PCOs del Comprobador Inferior del Juego de Pruebas, requieren una cabecera que anteceda a los parámetros de la primitiva; esta cabecera no aparece en la definición de los tipos de datos correspondientes en el Juego de Pruebas. Además, todos los mensajes hacen uso de la aplicación *HCTL Server*, y ésta requiere que el primer octeto sea una etiqueta que indique el tipo HCI (comando, evento o datos). Para añadir las cabeceras y etiquetas se ha modificado el procesado de las primitivas en la función `send_msg` de la librería `simbsp.a`. En recepción, se identifica el PCO por donde se recibe la primitiva y se eliminan los campos correspondientes.

Para todas las ASP se han creado nuevas funciones de codificación y descodificación que utilicen las funciones modificadas de las librerías. Para usar las funciones de codificación es suficiente con añadir la opción de compilación `ASP_SPECIAL_ENCODING`; las llamadas a las funciones de descodificación han de ser

[7] Esta regla proporciona una codificación *Little Endian* para cadenas de bits.

SPICUM servicio de publicaciones

incluidas en el código generado[8]. Las funciones creadas se han incluido en la librería `libasp.a`. En esta librería también se han incluido funciones auxiliares empleadas por las modificaciones realizadas en las librerías de la herramienta. Igualmente se han incluido las funciones que trabajan con cadenas de texto, pues deben mantener el formato de codificación *Big Endian*.

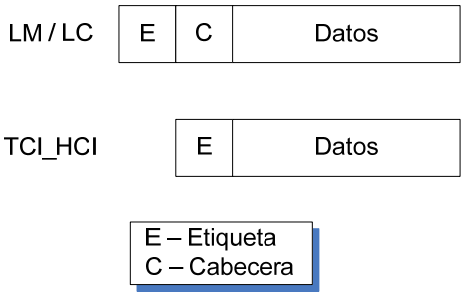
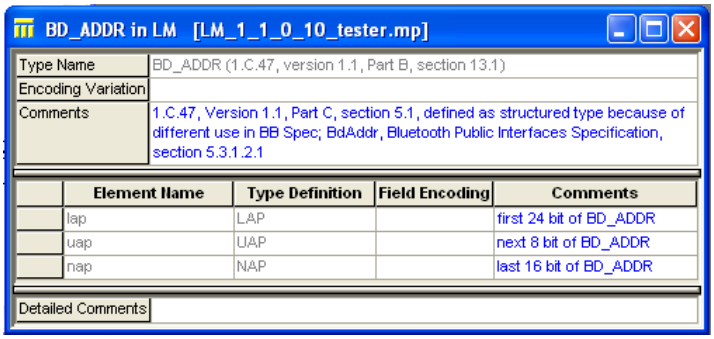


Figura 9.12: Campos que deben anteceder a la información intercambiada con el Subsistema de Pruebas.

En el código generado hay que modificar los ficheros `TYPE_FUNCTIONS<n>.C` cada vez que se regenere el código. Para incluir las modificaciones automáticamente se ha creado una macro, `type_functions.mac`, del editor de texto *UltraEdit* [ULTEDT]. Esta macro cambia el nombre de las funciones de descodificación incluidas en la librería `libasp.a` y sustituye las reglas de codificación por codificación directa donde corresponda.

La carga de los Parámetros de Pruebas también ha sido modificada, ya que el código generado no admite tipos estructurados, como es el caso del tipo de datos `BD_ADDR`. Este tipo de datos representa una dirección Bluetooth y contiene tres campos (Figura 9.13) de tipo `OCTETSTRING` de longitud 3, 1 y 2 octetos respectivamente. Se han modificado la función `init_tsptsv()`, que carga el valor de los Parámetros de Pruebas, y la función `init_varpar`, que guarda el valor leído en la variable correspondiente. La modificación en la función `init_varpar` hay que hacerla para cada Parámetro de Pruebas.


BD_ADDR in LM [LM_1_1_0_10_tester.mp]

Type Name	BD_ADDR (1.C.47, version 1.1, Part B, section 13.1)
Encoding Variation	
Comments	1.C.47, Version 1.1, Part C, section 5.1, defined as structured type because of different use in BB Spec; BdAddr, Bluetooth Public Interfaces Specification, section 5.3.1.2.1

	Element Name	Type Definition	Field Encoding	Comments
	lap	LAP		first 24 bit of BD_ADDR
	uap	UAP		next 8 bit of BD_ADDR
	nap	NAP		last 16 bit of BD_ADDR

Detailed Comments	

Figura 9.13: Definición del tipo de datos BD_ADDR.

[8] No existe la posibilidad de emplear una variable de compilación para que se utilicen unas funciones de descodificación propias.

SPICUM servicio de publicaciones

9.3.2.2 Pruebas

Para comprobar el comportamiento del Subsistema de Pruebas se han diseñado un emulador del Subsistema Inferior y un emulador de la Implementación Bajo Prueba. Estos emuladores han servido para verificar el modelado de los Casos de Prueba, detectar errores de la herramienta *TTCN Toolbox* y encontrar fallos en las modificaciones realizadas a las librerías y al código generado.

Las Pruebas de Sistema se han realizado sobre la IUT comercial. La configuración utilizada ha sido la mostrada en la Figura 9.14.

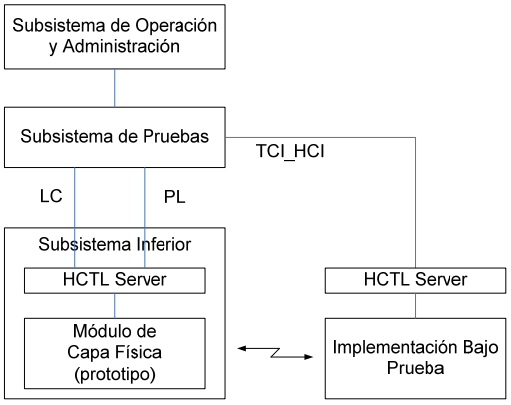


Figura 9.14: Configuración para las Pruebas de Sistema del Sistema de Pruebas para LM.

Como resultado de la evaluación realizada se puede decir que, en esencia, ambas herramientas (*Tau Suite* y *TTCN Toolbox*) poseen una complejidad similar, aunque las dificultades encontradas han sido diferentes en cada una de ellas. El principal inconveniente de *TTCN Toolbox* ha sido la carencia de soporte de los tipos de datos ASN.1. Además, aunque ofrecía el equivalente a los Módulos Adaptador y de Gestión de las Pruebas, ha sido necesario familiarizarse con ellos y modificarlos. Como principal punto fuerte se encuentra el hecho de que ofrece una mayor variedad y flexibilidad en los codificadores de la interfaz con el Subsistema Inferior. En resumen, al ya disponer de un entorno integrado para el diseño usando *Tau Suite*, no se consideró ventajoso el cambio a la nueva herramienta por el esfuerzo adicional que conllevaba.

9.4 Sistema de Pruebas de Interoperatividad

Con objeto de demostrar que es posible utilizar la arquitectura de la Metodología tanto para Sistemas de Pruebas de conformidad como de interoperatividad, se ha diseñado un prototipo de Sistema de Pruebas de interoperatividad para el perfil *Headset* [MORI03]. Las Especificaciones de Prueba de Bluetooth para este perfil sólo describen las pruebas a realizar, pero no proporcionan un Juego de Pruebas en TTCN, ya que es uno de los perfiles para los que no se contempla la necesidad de unas pruebas de conformidad.

El perfil *Headset* [BT HEAD] permite al usuario establecer conexiones de audio empleando dispositivos de manos libres. El perfil define dos roles: auricular (HS – *Headset*)[9], que corresponde al dispositivo manos libres, y pasarela de audio (AG – *Audio Gateway*), que corresponde al dispositivo Bluetooth a través del cual el auricular

[9] Normalmente se utiliza la denominación auricular para denotar a una equipo que combina auriculares y micrófono.

SPICUM servicio de publicaciones

establece la conexión. La pasarela de audio puede actuar como dispositivo terminal o como dispositivo intermedio, por ejemplo en el caso de un teléfono móvil que permite hablar mediante un dispositivo manos libres. El perfil *Headset* emplea los protocolos L2CAP, SDP y RFCOMM. La señalización hace uso de comandos AT[10] ([ATCOM]).

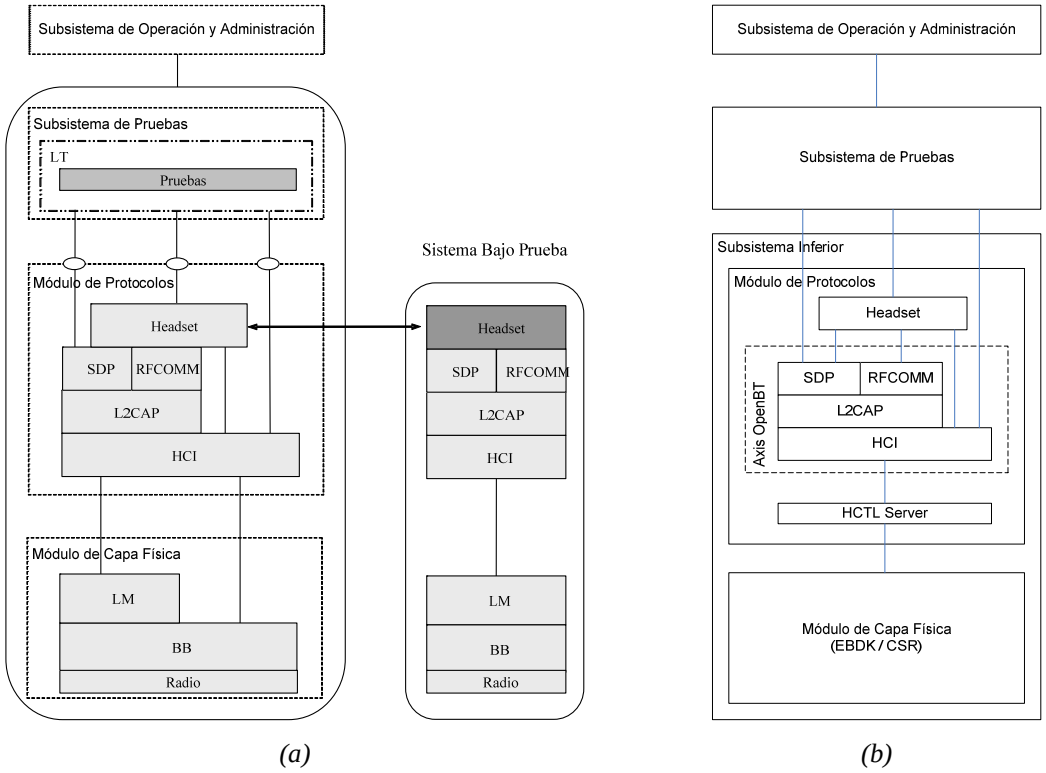


Figura 9.15: (a) Asignación de funcionalidad a cada Subsistema del Sistema de Pruebas para Headset y (b) Implementación de cada Subsistema.

Para este perfil se definen 17 Casos de Prueba [BTEST HEAD], cada uno de los cuales se puede ejecutar sobre una IUT actuando en el rol de auricular o en el rol de pasarela de audio; por tanto, hay un total de 34 Casos de Prueba diferentes. Estas Pruebas se organizan en cinco grupos que verifican las funcionalidades de establecimiento, liberación y transferencia de conexiones de audio, el modo *Park* y el control remoto del volumen.

A alto nivel, la arquitectura de este Sistema de Pruebas (Figura 9.15-a) es la misma que la de los Sistemas de Pruebas de conformidad. Sin embargo, su implementación difiere en algunos aspectos. La Figura 9.15-b muestra cómo se ha implementado cada Subsistema. El Subsistema de Pruebas se ha desarrollado en C, ya que uno de los objetivos perseguidos era que el usuario del Sistema de Pruebas pudiera diseñar sus propios Casos de Prueba, modificando los proporcionados o partiendo de cero.

El Módulo de Protocolos se ha implementado con la pila de protocolos Axis OpenBT [AXIS01] sobre la que se ha incorporado la funcionalidad del perfil *Headset*. Sobre la interfaz ofrecida se han programado las Pruebas en C. La pila Axis OpenBT es una implementación de libre distribución que incluye los protocolos L2CAP, SDP,

[10] AT significa *ATtention*. Los comandos AT son combinaciones de cadenas de texto que agrupadas producen comandos completos. El subconjunto de comandos AT utilizados está basado en [V.250].

SPICUM servicio de publicaciones

RFCOMM y TCS y la interfaz HCI. La elección de esta pila y no de la implementación descrita en la Sección 9.2.1 se debe a que en el momento de desarrollar este Sistema de Pruebas no se disponía de un modelo SDL de todos los protocolos necesarios, aunque fueron realizados posteriormente.

Al igual que los Sistemas de Pruebas anteriores, este Sistema de Pruebas se ha construido sobre la plataforma Windows 2000 y se ha utilizado el compilador Cygwin. Como Módulo de Capa Física se han empleado el dispositivo de CSR y el kit de desarrollo EBDK de Ericsson; como IUT se ha empleado un dispositivo comercial.

9.4.1 Módulo de Protocolos

El perfil *Headset* se ha implementado en C [MORI02a][11] y consiste en un conjunto de funciones básicas que ofrecen el servicio esperado. La interfaz con los niveles superiores se ha construido siguiendo la misma estrategia que emplea la pila Axis OpenBT. Existe una función a la que se le pasa como parámetro una estructura de datos que indica las funciones del nivel superior que deben ser invocadas cuando se produce cada uno de los posibles eventos.

La pila de protocolos Axis OpenBT v0.0.8 ha sido portada a Windows, pues la distribución sólo funcionaba en Linux. Está organizada en dos entidades, una implementa todos los niveles de protocolos y el cliente del protocolo SDP, la otra implementa el servidor SDP y se ejecuta como un hilo de la primera. En Linux, la pila se puede ejecutar en modo usuario y en modo *kernel*, que permite acceder a servicios de bajo nivel del sistema operativo. El portado a Windows se ha realizado sobre el código del modo usuario por ser el otro demasiado dependiente de la plataforma.

Tabla 9.3: Pasos de Prueba implementados en el Subsistema de Pruebas del perfil Headset.

Paso de Prueba	Descripción
`Connect`	Inicia una conexión RFCOMM con el IUT.
`Disconnect`	Inicia la desconexión del enlace RFCOMM.
`Add SCO`	Añade un enlace SCO a una conexión RFCOMM.
`Free SCO`	Libera un enlace SCO.
`RING`	Manda el comando AT RING.
`AT Command`	Construye un comando AT y lo envía.
`OK`	Envía un código OK como respuesta a un comando AT.
`ERROR`	Envía un código ERROR como respuesta a un comando AT.
`Park`	Aparca un dispositivo con el que se ha establecido un enlace RFCOMM.
`Unpark`	Desaparca un dispositivo aparcado previamente.
`SDP_CONN`	Inicia una conexión SDP para poder realizar peticiones.
`SDP_DISC`	Finaliza una conexión SDP
`SS_REQ`	Hace una petición de búsqueda de servicios de SDP.
`SA_REQ`	Solicita el canal de un servicio.

La adaptación a Windows de la pila ha requerido algunos cambios:

- Modificar el acceso a los dispositivos del sistema para que se realice mediante *sockets*.

[11] Los detalles de la implementación se pueden consultar en [MORI02b].

SPICUM servicio de publicaciones

- Completar el protocolo SDP para que pueda actuar simultáneamente como cliente y servidor. También se ha creado en XML (*eXtensible Markup Language*) la base de datos de servicios.
- Implementar el procedimiento de emparejamiento (*pairing*) entre dos dispositivos.
- Incluir temporizadores de control para la operación de algunos protocolos, por ejemplo, para cuándo iniciar una retransmisión.
- Modificar el nivel L2CAP para que gestione el vencimiento de los temporizadores de una conexión, maneje retransmisiones y libere recursos tras la desconexión.

9.4.2 Subsistema de Pruebas

El Subsistema de Pruebas se ha modelado en C, construyendo una función por cada Caso de Prueba (y rol de la IUT). Además, hay funcionalidades que se han modelado como Pasos de Prueba (Tabla 9.3) y que, aparte de ser empleadas en los Casos de Prueba, también están a disposición del usuario que quiera diseñar sus propias Pruebas. Los Casos de Prueba se ejecutan en un hilo diferente del programa principal para que el usuario pueda continuar interaccionando con el sistema.

9.4.3 Pruebas

Se han realizado pruebas a cada uno de los componentes construidos y a las sucesivas integraciones. La pila Axis OpenBT ha sido probada inicialmente en Linux con objeto de determinar su funcionalidad. Estas pruebas se han realizado en primera instancia sin *Host Controller*, es decir, sólo con los protocolos desde la interfaz HCI hacia arriba (Figura 9.16-a); la pila dispone de una emulación a nivel HCI que permite la conexión de dos pilas local o remotamente. Para las pruebas con un Módulo de Capa Física se han empleado el *Host Controller* EBDK bajo la pila y un *Host Controller* comercial como IUT, aunque éste sólo incluye la funcionalidad hasta la interfaz HCI.

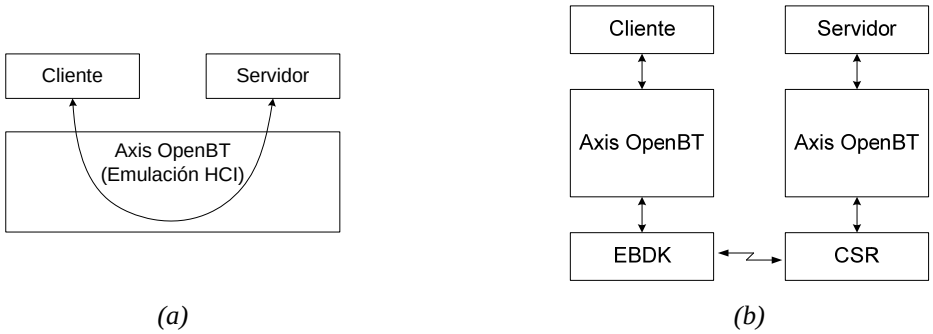


Figura 9.16: Pruebas de la funcionalidad de la pila Axis OpenBT con (a) emulación HCI y (b) dispositivos reales.

Una vez portada a Windows se han repetido las pruebas anteriores en la nueva plataforma. En este caso se ha utilizado la configuración anterior de *Host Controllers* y también una configuración con dos *Host Controller* de CSR[12]. En ambos extremos se

[12] En Linux no es posible utilizar un *Host Controller* de CSR bajo la pila porque la aplicación *HCTL Server* sólo funciona en Windows.

SPICUM servicio de publicaciones

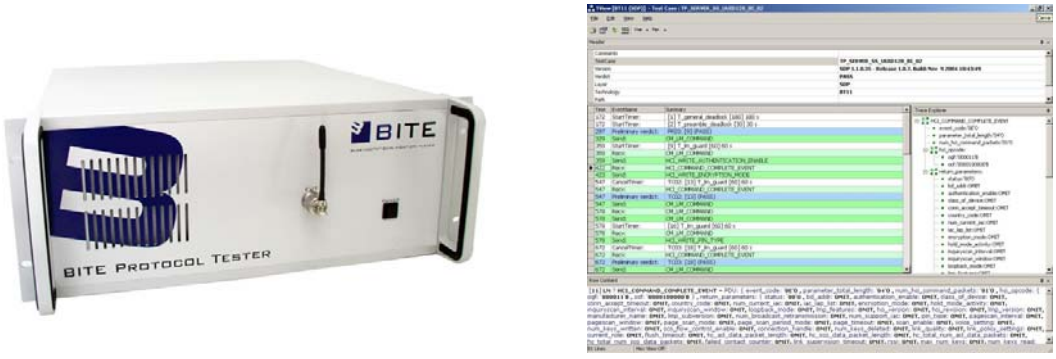
ejecuta la pila Axis OpenBT. También se han realizado al servidor SDP parte de las pruebas especificadas para él (Figura 9.16-b).

Las Pruebas de Subsistema se han realizado al mismo tiempo que se probaba el perfil *Headset*. Para ello, se ha ejecutado una instancia del Subsistema Inferior en dos PCs, cada uno de ellos actuando en un rol distinto, auricular o pasarela. Como Módulo de Capa Física es posible emplear indistintamente el *Host Controller* EBDK o el CSR. Posteriormente se ha enfrentado el Subsistema Inferior actuando en ambos roles con un dispositivo *Headset* y una pasarela (un teléfono móvil) de Ericsson.

Las Pruebas de Sistema se han realizado sobre los dispositivos indicados de Ericsson. Estas pruebas han servido también para comprobar el comportamiento del Subsistema de Pruebas.

9.5 Sistemas Comerciales

Sólo se han comercializado dos Sistemas de Pruebas de protocolos para Bluetooth desarrollados, respectivamente, por AT4 wireless (BITE) y Rohde & Schwarz (PTW60). El Sistema de Pruebas BITE [BITE], construido por AT4 wireless (Figura 9.17-a) usando la Metodología desarrollada en esta Tesis, ha dominado el mercado. Este Sistema de Pruebas se puede utilizar con las distintas versiones de la tecnología Bluetooth. Permite la certificación de la conformidad de los protocolos y perfiles BB, LM, HCI, L2CAP, SDP, RFCOMM, SPP y GAP e incorpora la funcionalidad necesaria para la realización de pruebas, su gestión y la generación de informes de forma automática (Figura 9.17-b).



(a) *(b)*

Figura 9.17: Sistema de Pruebas de protocolos BITE: (a) Equipo y (b) Subsistema de Operación y Administración.

9.6 Conclusiones

En este capítulo se han presentado distintas actividades realizadas alrededor de la tecnología Bluetooth. La más importante ha sido la aplicación de la Metodología de Diseño en un entorno industrial participando en la construcción de un Sistema comercial de Pruebas de protocolos. La colaboración ha consistido en el diseño del Módulo de Protocolos así como en el uso de distintas herramientas asociadas a la Metodología, tanto componentes como generadores automáticos, lo que ha permitido evolucionar las herramientas y componentes empleados en los Sistemas de Pruebas para

SPICUM servicio de publicaciones

DECT. También se ha implementado una pila completa de protocolos, aunque no se ha descrito en este capítulo, y se ha participado en la validación de los Juegos de Pruebas oficiales de distintos protocolos Bluetooth .El Sistema de Pruebas BITE ha sido todo un éxito, convirtiéndose en el sistema de referencia para la tecnología Bluetooth.

Se ha demostrado cómo la arquitectura propuesta para los Sistemas de Pruebas permite el uso de distintas herramientas para la implementación de cada Subsistema. En concreto, se han construido los Subsistemas de Pruebas para los niveles LM y SPP con una herramienta alternativa. Se ha visto que el uso de una u otra herramienta ofrece una complejidad similar a la hora de realizar la integración con el Subsistema Inferior. Las adaptaciones necesarias han venido por el lado de la codificación de la interfaz con el Subsistema Inferior; para parte de estas adaptaciones se han generado herramientas automáticas. Aunque el código generado por la herramienta evaluada en este capítulo es más legible, la carencia de soporte para los tipos ASN.1 es algo importante dada la evolución previsible en esta época para el diseño de los Juegos de Pruebas.

Además, se ha empleado la misma arquitectura para un prototipo de un Sistema de Pruebas de interoperatividad del perfil *Headset*. El sistema resultante permite al usuario la modificación de las pruebas y la creación de nuevos escenarios. El Módulo de Protocolos se ha basado en una pila de libre distribución sobre la que se ha implementado la funcionalidad del perfil, lo que demuestra la posibilidad de emplear lenguajes distintos a los sugeridos para el diseño de los Subsistemas de la Arquitectura.

SPICUM servicio de publicaciones

Capítulo

10

"Y allá, en la lejana Tierra, el doctor Carlisle Perera no le habla confiado todavía a nadie que despertó de un sueño agitado con un mensaje de su subconsciente resonando en su cerebro: Los ramanes lo hacen todo por triplicado."

Carlisle Perera, Cita con Rama

CAPÍTULO 10: SISTEMAS DE PRUEBAS UMTS

A diferencia de las tecnologías DECT y Bluetooth, que pertenecen a la categoría de acceso inalámbrico, UMTS (*Universal Mobile Telecommunications System*) forma parte de los sistemas de acceso móviles donde la infraestructura de red pertenece a un operador con licencia pública. Esto lo convierte en un sistema con un plano de señalización notablemente más complejo que los anteriores, ya que debe acomodar mecanismos que otorguen el acceso a un notablemente mayor número, y una mayor variedad, de usuarios en entornos abiertos. El conjunto de servicios ofrecidos es muy diverso, y el diseño del sistema debe ser capaz de servir a la demanda de los mismos de una forma eficiente en el uso de los recursos, tanto para el usuario como para el operador.

La filosofía de diseño de los Juegos de Pruebas estandarizados por 3GPP ha evolucionado en algunos aspectos. Uno de estos puntos son los Métodos de Pruebas de los distintos protocolos, para lo que se ha definido una única arquitectura que se utiliza para todas las Pruebas, con la posibilidad de emplear módulos adicionales cuando es necesario. Además, se ha adoptado la notación ASN.1 como la preferida para definir las primitivas y unidades de datos, lo que facilita la reutilización de los mensajes del Nivel 3 definidos en las Especificaciones de Sistema. Esta posibilidad existía en TTCN desde el principio pero no había sido realmente explotada hasta el momento. Esto ha exigido extender los generadores automáticos de interfaces de que ya se disponía.

UMTS, visto como la evolución de GSM, ofrece un aumento considerable en la velocidad del acceso radio. Esta mayor velocidad requiere nuevas soluciones en la interacción entre el Módulo de Capa Física, de operación muy rápida, y el resto del Sistema de Pruebas, cuya operación requiere una velocidad cada vez menor conforme se sube de nivel. La modularidad de la arquitectura empleada facilita el diseño, permitiendo que cada componente se ejecute en la plataforma más adecuada en términos de esfuerzo de diseño y costes.

SPICUM servicio de publicaciones

La Metodología de Diseño ha sido aplicada en la construcción de un Sistema comercial de Pruebas de conformidad para protocolos (MINT) ([PROY02], [PROY03], [CONT02], [CONT03a]). En concreto, ha habido una implicación directa en el diseño del Módulo de Protocolos ([COLA02], [COLA03], [SORE03b]). Además, se ha avanzado en la formalización del diseño de Sistemas de Pruebas de interoperatividad, integrando herramientas comerciales en el proceso y adaptando las herramientas propias [MORI04]. El resultado de la colaboración ha sido también utilizado como Unidad de Señalización[1] de un Sistema de Pruebas de conformidad para radio.

En este capítulo se describen aquellos aspectos que constituyen un avance respecto al diseño de los Sistemas de Pruebas Bluetooth descrito en el capítulo anterior[2]. En primer lugar, se presenta la tecnología UMTS, con objeto de explicar los conceptos necesarios para comprender el resto del capítulo. A continuación, se describe el diseño del Sistema de Pruebas de conformidad para protocolos. Primeramente, se exponen las ideas en que se ha basado el diseño del sistema y posteriormente se describen las interfaces empleadas por el Módulo de Protocolos y el modelado de cada uno de los Niveles que contiene. Seguidamente, se explica el diseño realizado para el Sistema de Pruebas de interoperatividad. Por último, se resumen algunos sistemas que se han comercializado para esta tecnología.

10.1 Descripción del Sistema UMTS

UMTS es la rama europea de la familia de tecnologías de comunicaciones inalámbricas de tercera generación IMT-2000 [3GPP]. Fue diseñada para ofrecer un ancho rango de servicios de voz, datos y multimedia, con velocidades de hasta 2 Mbps (Tabla 10.1), ofreciendo prestaciones avanzadas como negociación de la calidad de servicio, seguridad o itinerancia mundial. La banda de frecuencias habilitada para estos sistemas se muestra en la Figura 10.1. Utiliza la tecnología de acceso radio WCDMA (*Wideband Code Division Multiple Access*), con un ancho de banda de 5 MHz por portadora. En el modo FDD (*Frequency Division Duplex*) utiliza dos bandas disjuntas de 5 MHz cada una, y en el modo TDD (*Time Division Duplex*) una misma banda para ambos sentidos de la comunicación.

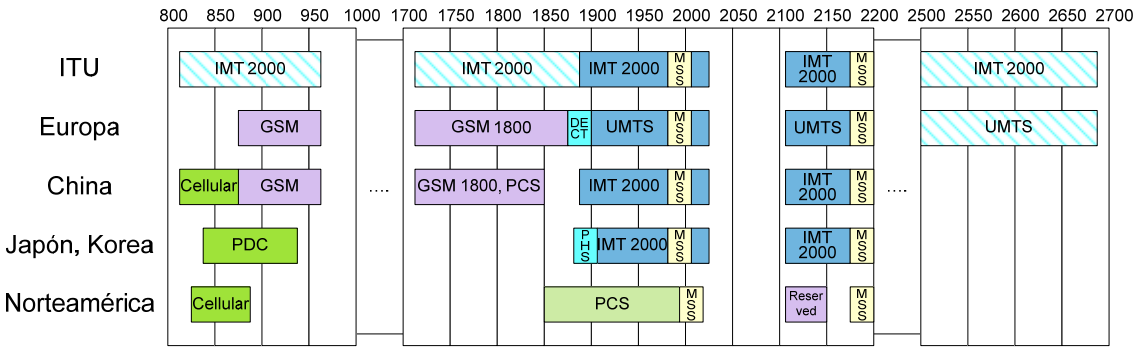


Figura 10.1: Asignación del espectro radioeléctrico a distintas tecnologías inalámbricas [WRC00].

[1] La Unidad de Señalización es la encargada de situar al Equipo Bajo Prueba en el estado adecuado para realizar la prueba radio.

[2] Otros trabajos realizados sobre la tecnología UMTS se pueden encontrar en [ALAR01], [BLAN02], [IZQU03] y [RODR04].

SPICUM servicio de publicaciones

Tabla 10.1: Tasas de transferencia según el entorno y la movilidad del usuario.

Área	Velocidad Máxima de Datos (kbps)	Velocidad Máxima del Móvil (km/h)	Tipo de Celda
Rural	144	500	Macrocelda
Suburbana	384	20	Macrocelda Microcelda
Interior	2048	10	Microcelda Picocelda

El sistema UMTS consta de tres componentes principales (Figura 10.2) [3GPP 23.002]: equipos móviles, denominados equipos de usuario (UE – *User Equipment*); red de acceso radio, denominada UTRAN (*UMTS Terrestrial Radio Access Network*); y la red fija del sistema, denominada Núcleo de Red (CN – *Core Network*). La red de acceso radio, UTRAN, se subdivide a su vez en Nodos B, que son los responsables de la transmisión entre el UE y la UTRAN, y RNCs (*Radio Network Controller*), que gestionan los recursos radio. Son los RNC los que proporcionan al sistema las nuevas funcionalidades como son el uso de macrodiversidad o la gestión de la movilidad. La red del núcleo, CN, contiene aquellas entidades que proporcionan soporte a la red de acceso y los servicios de telecomunicación prestados en la misma [3GPP 23.101]. Estas entidades son: HLR (*Home Location Register*), AuC (*Authentication Center*) y EIR (*Equipment Identity Register*), que almacenan información para gestionar y autentificar a los usuarios de la red, MSC (*Mobile Switching Center*) y VLR (*Visitor Location Register*), que manejan servicios de conmutación de circuitos, GMSC (*Gateway Mobile Switching Center*), que encamina la llamada hacia el MSC en el que se encuentra el usuario llamado, y SGSN (*Serving GPRS Support Node*) y GGSN (*Gateway GPRS Support Node*), que proporcionan servicios de conmutación de paquetes. Las interfaces entre cada uno de los componentes se muestran en la figura.

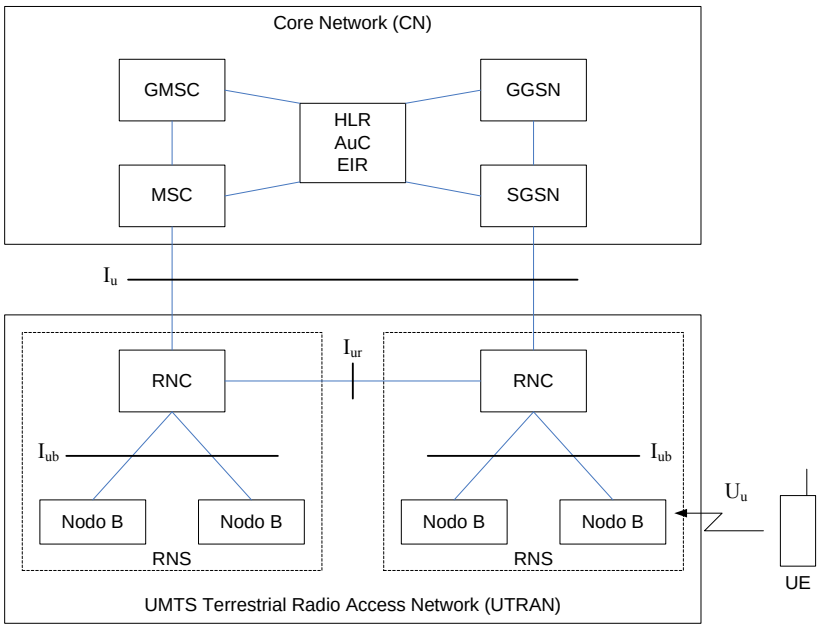


Figura 10.2: Entidades que forman el sistema UMTS.

SPICUM servicio de publicaciones

La arquitectura de protocolos de la interfaz radio se muestra en la Figura 10.3 [3GPP 25.301]. Desde el punto de vista de la estructura de protocolos la arquitectura UMTS se divide en dos estratos: de acceso (AS – *Access Stratum*), que realiza la transferencia de información en la interfaz aire, y de no-acceso (NAS – *Non-Access Stratum*), que incluye los protocolos que transportan los datos entre el usuario y el núcleo de la red. Nuestro interés reside en el estrato de acceso. Los Niveles RRC (*Radio Resource Control*) y RLC (*Radio Link Control*) se dividen en plano de control (*C-plane*), que maneja los mensajes de señalización, y plano de usuario (*U-plane*), que maneja la información de usuario; los Niveles PDCP (*Packet Data Convergence Protocol*) y BMC (*Broadcast/Multicast Control*) sólo existen en el plano de usuario. En el plano de control se transmiten, fundamentalmente, dos tipos de mensajes de señalización sobre la interfaz radio: mensajes del Nivel RRC y mensajes de los niveles superiores (NAS). Se denomina Portadora de Acceso Radio (RAB – *Radio Access Bearer*) al servicio que proporciona el *Access Stratum* al NAS para la transferencia de datos de usuario entre el UE y el Núcleo de Red.

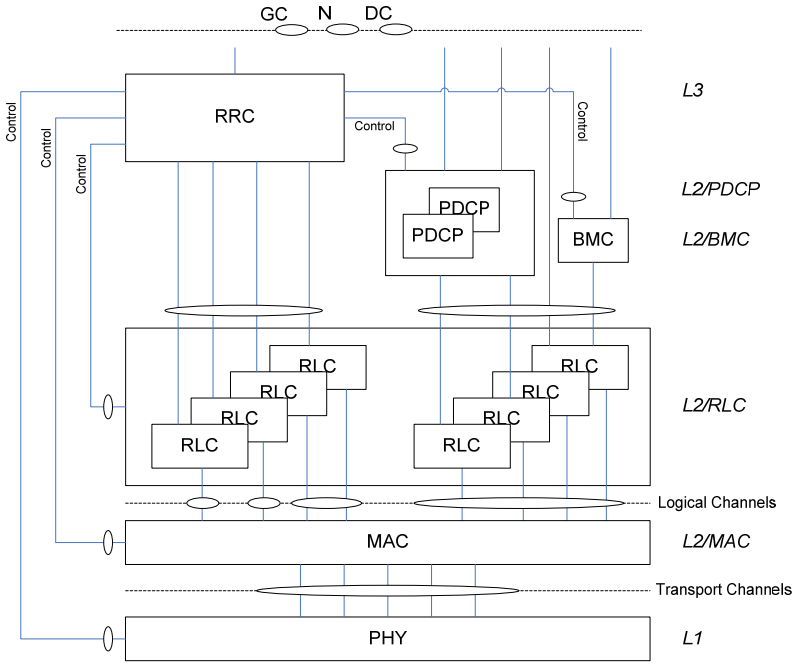


Figura 10.3: Arquitectura de la interfaz radio.

El Nivel Físico (L1 o PHY – *PHYsical layer*) [3GPP 25.201] define las características de transferencia del servicio ofrecido a través de los canales de transporte. La información de estos canales es multiplexada en uno o más canales físicos antes de ser transmitida por la interfaz aire. Los canales de transporte se dividen en comunes, donde se identifica al UE destino u origen de la información, y dedicados, donde el UE se identifica por el canal físico (código + frecuencia en FDD). La lista de canales de transporte se muestra en la Tabla 10.2. Las características de un canal de transporte están definidas por su formato de transporte, en el que se especifican aspectos como el tamaño de los bloques de transporte, la codificación convolucional a emplear o el entrelazado (ver Sección 10.1.1). La sincronización del Nivel Físico establece que se transmita una trama con una periodicidad (TTI – *Transmission Time Interval*) que depende de la velocidad de

SPICUM servicio de publicaciones

transferencia solicitada (2/10/20/40/80 ms); cada trama agrupa uno o más bloques de transporte.

Tabla 10.2: Tipos de canales de transporte [TORR02].

	Sigla	Nombre	Sentido	Descripción
Comunes	RACH	*Random Access CHannel*	↑	Peticiones de conexión o envío de pequeñas cantidades de datos.
	CPCH	*Common Packet CHannel*	↑	Transmisión de datos en modo paquete (sólo en FDD)
	FACH	*Forward Access CHannel*	↓	Transmisión de información de control y envío de pequeñas cantidades de datos (por ejemplo, respuesta a solicitud de acceso aleatorio)
	DSCH	*Downlink Shared CHannel*	↓	Información de control dedicado o datos.
	BCH	*Broadcast CHannel*	↓	Difusión de la información del sistema en una celda
	PCH	*Paging Channel*	↓	Información de búsqueda (*paging*) de un terminal.
Dedicado	DCH	*Dedicated CHannel*	↑↓	Transporta datos de usuario.

El Nivel Físico se encarga de: multiplexar los canales de transporte, combinar canales físicos, modular/remodular los canales físicos, sincronizar en frecuencia y tiempo (chip, bit, ranura, trama), controlar la potencia de bucle cerrado, realizar la codificación y descodificación FEC (*Forward Error Correction*) en los canales de transporte, realizar medidas y comunicarlas a los niveles superiores (FER – *Frame Error Rate*, SIR - *Signal-to-Interference Ratio*, BER – *Bit Error Rate*, potencia interferente, potencia de transmisión, etc.), detectar errores en los canales de transporte, etc.

El Nivel de Enlace (L2) está dividido en varios Subniveles: Control de Acceso al Medio (MAC – *Medium Access Control*), Control del Enlace Radio (RLC – *Radio Link Control*), Protocolo de Convergencia de Paquetes de Datos (PDCP – *Packet Data Convergence Protocol*) y Control de Difusión/Multidifusión (BMC - *Broadcast/Multicast Control*).

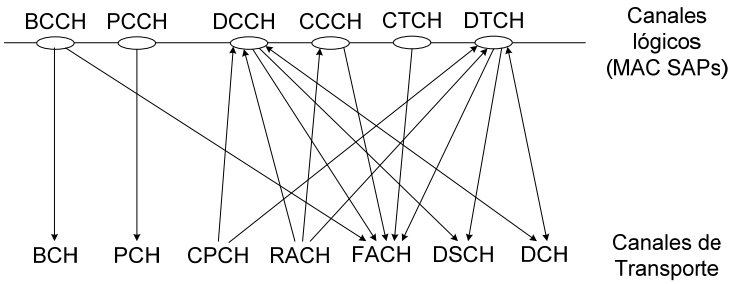


Figura 10.4: Correspondencia entre canales lógicos y canales de transporte en la UTRAN.

El Nivel MAC [3GPP 25.321] ofrece un servicio de transferencia de datos a través de canales lógicos, asigna recursos radio en función de la configuración realizada por el Nivel RRC y proporciona medidas durante la operación. Las características del servicio

SPICUM servicio de publicaciones

de datos están definidas por el tipo de canal lógico empleado. Los canales lógicos se dividen (Tabla 10.3) en canales de control, que se utilizan para transferir datos del plano de control, y canales de tráfico, que transportan datos del plano de usuario. El Nivel MAC se encarga de realizar la correspondencia/multiplexación entre los canales lógicos y los canales de transporte (Figura 10.4).

Tabla 10.3: Tipos de canales lógicos.

	Sigla	Nombre	Sentido	Descripción
Control	BCCH	*Broadcast Control CHannel*	↓	Difusión de información del sistema.
	PCCH	*Paging Control CHannel*	↓	Información de búsqueda (*paging*).
	DCCH	*Dedicated Control CHannel*	↑↓	Transmisión punto a punto de información de control de/para un UE.
	CCCH	*Common Control CHannel*	↑↓	Información de control para UEs que no están en modo conectado.
	SHCCH	*SHared channel Control CHannel*	↑↓	Información de control para canales compartidos (sólo FDD).
Tráfico	DTCH	*Dedicated Traffic CHannel*	↑↓	Canal punto a punto para datos de usuario.
	CTCH	*Common Traffic CHannel*	↑↓	Canal punto a multipunto (grupo de UEs) para datos de usuario.

La arquitectura descrita en la norma para el Nivel MAC es la mostrada en la Figura 10.5. La entidad MAC-c/sh gestiona la transferencia de datos sobre canales de transporte compartidos y comunes, mientras que la entidad MAC-d se encarga de los canales de transporte dedicados. La tercera entidad, MAC-b, es responsable del canal de transporte de difusión.

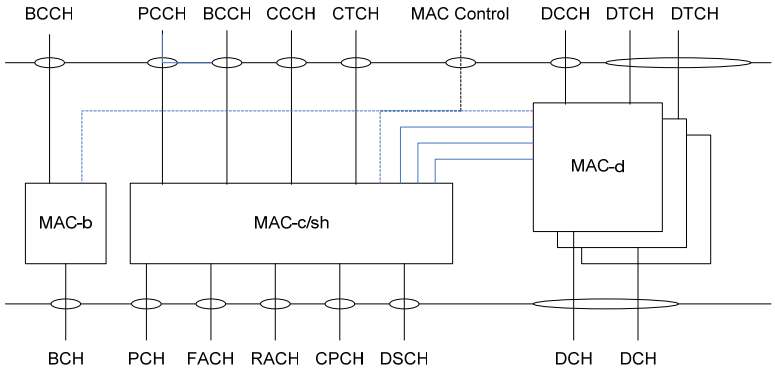


Figura 10.5: Arquitectura del Nivel MAC en la UTRAN (FDD)

El Nivel MAC es responsable de realizar la correspondencia entre canales lógicos y canales de transporte, seleccionar el formato de transporte, gestionar las prioridades entre flujos de datos, multiplexar PDUs de niveles superiores, cifrar la información[3] y monitorizar los volúmenes de tráfico.

[3] Sólo en el modo transparente de RLC.

SPICUM servicio de publicaciones

El Nivel RLC [3GPP 25.322] ofrece tres tipos de servicios de transferencia de datos (Figura 10.6): transparente (TM – *Transparent Mode*), no confirmado (UM – *Unacknowledged Mode*) y confirmado (AM – *Acknowledged Mode*).

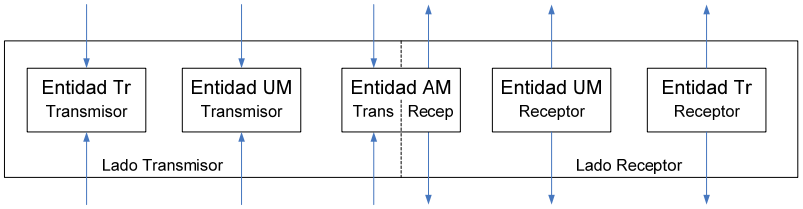


Figura 10.6: Entidades del Nivel RLC.

El concepto básico en este Nivel es el concepto de Portadora Radio (RB – *Radio Bearer*), que define un servicio para transmitir datos entre el UE y el RNC que le da servicio. Durante la operación se crean en RLC una o más entidades del mismo o diferente tipo; cada entidad creada se asocia a una única portadora radio. Las portadoras radio del plano de control, que sirven a RRC, se denominan portadoras radio de señalización. La portadora radio de señalización RB0 siempre se encuentra establecida. Al establecerse la conexión de señalización se crean 3 ó 4 portadoras radio de señalización (Tabla 10.4) en modo UM o AM; tras el establecimiento se pueden crear nuevas portadoras radio de señalización que empleen el modo transparente.

Tabla 10.4: Portadoras radio creadas al establecer la conexión de señalización.

RB	Canal Lógico	Descripción
RB0	CCCH	Siempre está establecida.
RB1	DCCH	Modo UM.
RB2	DCCH	Modo AM salvo mensajes de señalización RRC con señalización NAS.
RB3	DCCH	Modo AM para mensajes de señalización NAS de alta prioridad.
RB4	DCCH	Modo AM para mensajes de señalización NAS de baja prioridad.

Entre las funciones del Nivel RLC se encuentran la segmentación y reensamblado de PDUs de niveles superiores en bloques de transporte (PUs RLC – *Payload Units*), la concatenación de SDUs (*Service Data Unit*) de RLC, la corrección de errores, el control de flujo (modo AM) y el cifrado (modos UM y AM).

Los Niveles PDCP [3GPP 25.323] y BMC [3GPP 25.324] pertenecen exclusivamente al plano de usuario. El Nivel PDCP se emplea en la conmutación de paquetes, y realiza la compresión de cabeceras para la interfaz aire. El Nivel BMC ofrece un servicio de difusión y multidifusión en el plano de usuario, y se emplea para datos comunes en modo transparente o no confirmado.

El Nivel de Control de los Recursos Radio (L3 o RRC – *Radio Resource Control*) es el encargado de controlar el plano de señalización. La configuración de los niveles inferiores la realiza a través de las interfaces de control con cada uno de ellos; estas interfaces son directas entre el Nivel RRC y el nivel correspondiente, violando la filosofía OSI. El Nivel RRC ofrece tres tipos de servicios a los niveles superiores: control general (GC – *General Control*), que ofrece un servicio de difusión en un área determinada; notificación (Nt – *Notification*), que ofrece un servicio de difusión de

SPICUM servicio de publicaciones

información de *paging* y de notificación; y control dedicado (DC – *Dedicated Control*), que ofrece un servicio de conexión y transferencia de mensajes. Todas las conexiones de señalización de un UE se multiplexan en una única conexión RRC.

El Nivel RRC [3GPP 25.331] está organizado en varias entidades (Figura 10.7): BCFE (*Broadcast Control Functional Entity*), que controla la difusión de información; PNFE (*Paging Notification Functional Entity*), que controla los mensajes de búsqueda (*paging*); DCFE (*Dedicated Control Functional Entity*), que controla las conexiones y los recursos asociados a las mismas, encargándose de su establecimiento, mantenimiento y liberación y la gestión de las claves de cifrado; y TME (*Transfer Mode Entity*), que realiza la multiplexación de mensajes RRC en primitivas RLC.

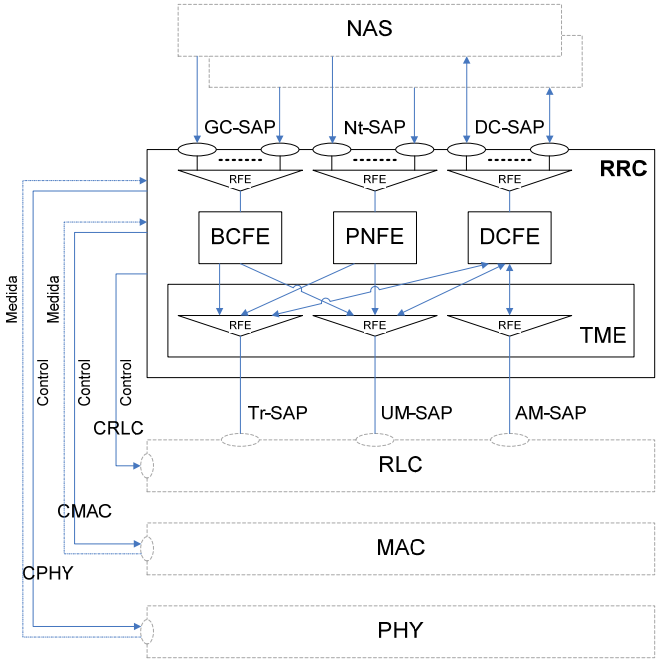


Figura 10.7: Estructura del Nivel RRC (para FDD).

Entre las funciones del Nivel RRC se encuentran la gestión de conexiones de señalización, el establecimiento, reconfiguración y liberación de portadoras radio, la gestión de recursos radio necesarios para las conexiones y los procedimientos de movilidad, el control del cifrado, la solicitud de informes de medida, y la selección y reselección de celda.

10.1.1 Formatos de Transporte

En UMTS la comunicación se adapta dinámicamente, cada TTI, a las condiciones del medio; el tipo de codificación empleada se define en un elemento denominado formato de transporte. El Nivel RRC es el responsable de indicar a los niveles inferiores el conjunto de posibles formatos de transporte que se pueden utilizar; esta información también es comunicada a la entidad receptora. El uso de un cierto formato de transporte en un canal tiene un impacto directo sobre la capacidad disponible para los demás canales, por lo que sólo ciertas combinaciones de formatos de transporte (uno de cada canal) son posibles en cada TTI. En cada TTI, el transmisor elige una combinación de entre las posibles, informando de dicha elección a la entidad receptora. A continuación

SPICUM servicio de publicaciones

se describen todos los elementos que se utilizan en este mecanismo de adaptación dinámica; para facilitar la comprensión, la Tabla 10.5 muestra un resumen de dichos elementos.

Tabla 10.5: Conceptos relacionados con la configuración y selección de formatos de transporte.

Sigla	Nombre	Descripción
TF	*Transport Format*	Formato de transporte. Cada canal de transporte utiliza un formato de transporte en un determinado TTI. Da características como el número de bloques de datos que se envían y de qué tamaño son.
TFS	*Transport Format Set*	Conjunto de formatos de transporte (TF) que tiene un canal de transporte.
TFI	*Transport Format Indicator*	Indicador de formato de transporte. Es un número que identifica a un TF concreto de entre todos los que tiene un canal de transporte.
TFSS	*Transport Format Set Set*	Conjunto de los conjuntos de formatos de transporte (TFS) de todos los canales de transporte.
TFC	*Transport Format Combination*	Combinación de formatos de transporte. Se trata de elegir un formato de transporte (TF) para cada canal de transporte.
TFCS	*Transport Format Combination Set*	Conjunto de combinaciones de formatos de transporte (TFC). Indica todas las combinaciones que se configuran.
TFCI	*Transport Format Combination Indicator*	Es un índice que indica la combinación usada de entre las presentes en el conjunto de combinaciones de formatos de transporte (TFCS).
CTFC	*Calculated Transport Format Combination*	Es un número que identifica unívocamente a una combinación de formatos de transporte (TFC) concreta.

Un formato de transporte (TF – *Transport Format*) está dividido en dos partes, una dinámica y una estática. En la parte dinámica se encuentran el tamaño del bloque de transporte (TB – *Transport Block*), el número de bloques de transporte y el TTI a utilizar. La parte estática la forman el tipo de protección de errores, la tasa de codificación y el tamaño del CRC. Cada vez que los Niveles PHY y MAC intercambian una primitiva de datos, se indica el formato de transporte a utilizar.

Cuando el Nivel RRC configura los canales de transporte se indican los posibles formatos de transporte en cada canal. El conjunto de formatos de transporte asociado a un canal se denota por TFS (*Transport Format Set*). Un canal de transporte puede tener un máximo de 32 formatos de transporte.

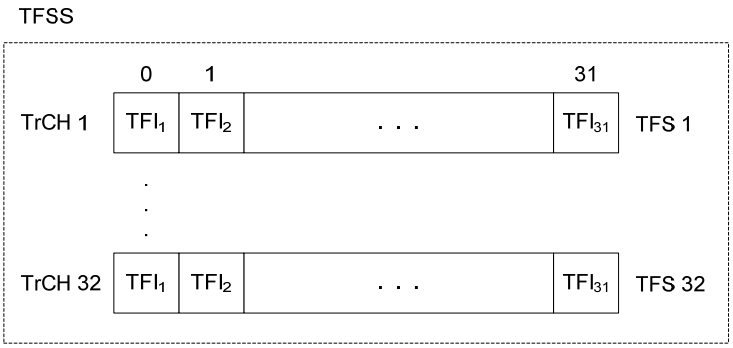


Figura 10.8: Conjunto de conjuntos de formatos de transporte (TFSS).

SPICUM servicio de publicaciones

Como puede haber hasta 32 canales de transporte, se pueden tener hasta 32 TFS, uno para cada canal. El conjunto de conjuntos de formatos de transporte se denomina TFSS (*Transport Format Set Set*) (Figura 10.8). Dentro de un conjunto de formatos de transporte, los formatos de transporte que lo forman se identifican a través de un número, el TFI (*Transport Format Identifier*), que es el índice que tiene asociado el formato de transporte dentro del conjunto (como elemento de un *array*).

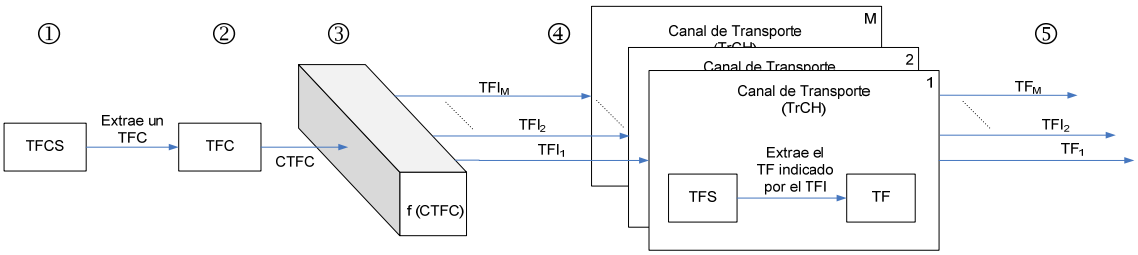


Figura 10.9: Proceso de selección de los formatos de transporte de cada canal.

Una combinación de formatos de transporte indica un formato de transporte para cada canal de transporte; los formatos de transporte se identifican con el TFI. Por tanto, una combinación de formatos de transporte puede estar formada por hasta 32 TFI. Por ejemplo, una combinación de formatos de transporte en el caso de tener tres canales de transporte podría ser: TFC = (1, 0, 3), indicando que hay que usar el TFI 1 para el primer canal, el TFI 0 para el segundo y el TFI 3 para el tercer canal.

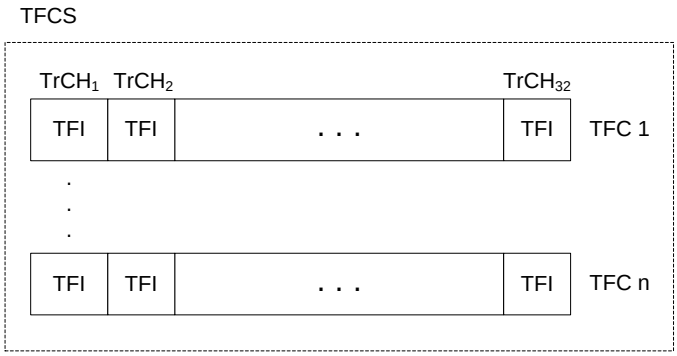


Figura 10.10: Conjunto de combinaciones de formatos de transporte.

La elección de qué formato de transporte emplear en un canal de transporte se realiza dinámicamente en el Nivel MAC cada TTI, aunque esta elección no es independiente de los formatos de transporte elegidos para el resto de canales (Figura 10.9). Sólo son posibles algunas combinaciones de formatos de transporte (TFC – *Transport Format Combination*). Durante el proceso de configuración se le indica al Nivel MAC el conjunto de combinaciones de formatos de transporte (TFCS – *Transport Format Combination Set*) que puede utilizar (Figura 10.10). Para no tener que enviar todos los índices que supone una combinación de formatos de transporte (hasta 32), cada combinación tiene asociado un número que lo identifica unívocamente, el CTFC (*Calculated Transport Format Combination*). El cálculo de este valor se describe en la Sección 14.10 de [3GPP 25.331]. De esta forma, el Nivel RRC comunica el conjunto de combinaciones de formatos de transporte enviando únicamente el CTFC de cada

SPICUM servicio de publicaciones

combinación incluida en el conjunto. Una vez comunicadas las posibles combinaciones, al seleccionar una de ellas se transmite el índice de la posición que ocupa en el conjunto (TFCI – *Transport Format Combination Indicator*).

Por tanto, la configuración de los formatos de transporte tiene dos partes: el conjunto de conjuntos de formatos de transporte (TFSS), que da la información de los formatos de transporte para cada canal de transporte, y el conjunto de combinaciones calculadas de formatos de transporte (conjunto de CTFC), que da la información de las combinaciones de formatos de transporte que se pueden utilizar.

10.2 Sistema de Pruebas de Conformidad

El Sistema de Pruebas de conformidad de protocolos, MINT (*Mobile communications INtegrated Tester*), utiliza la arquitectura propuesta por la Metodología y las herramientas asociadas a la misma. Como parte de la colaboración con AT4 wireless se ha diseñado el Módulo de Protocolos, el cual también se ha utilizado como parte de la Unidad de Señalización del Sistema de Pruebas de radio de la misma familia.

En UMTS existen Juegos de Pruebas de conformidad de protocolos para RLC, MAC, PDCP, BMC, RRC, SMS (*Short Message Service*) y NAS. Los Métodos de Pruebas (Apéndice C) que se han definido emplean el mismo Subsistema Inferior para todos los protocolos [3GPP 34.123-3], aunque esta arquitectura es particularizada en el caso de las pruebas de NAS y SMS (funcionalidad de transferencia directa de RRC) y BMC (emulación).

Las Pruebas utilizan 8 celdas, aunque sólo puede haber un máximo de 6 celdas activas simultáneamente. De estas celdas, dos de ellas actúan como la celda actual y la celda objetivo del UE en caso de traspaso, por lo que, en un instante dado, sólo dos celdas deben ofrecer toda la funcionalidad mientras que el resto de las celdas se limitan a emitir el canal de difusión. Para evitar que los canales dedicados se desconecten en caso de que se produzca un traspaso, se agrupan en una celda específica (celda dedicada) que el Sistema de Pruebas asocia automáticamente con la celda en la que se encuentre el UE.

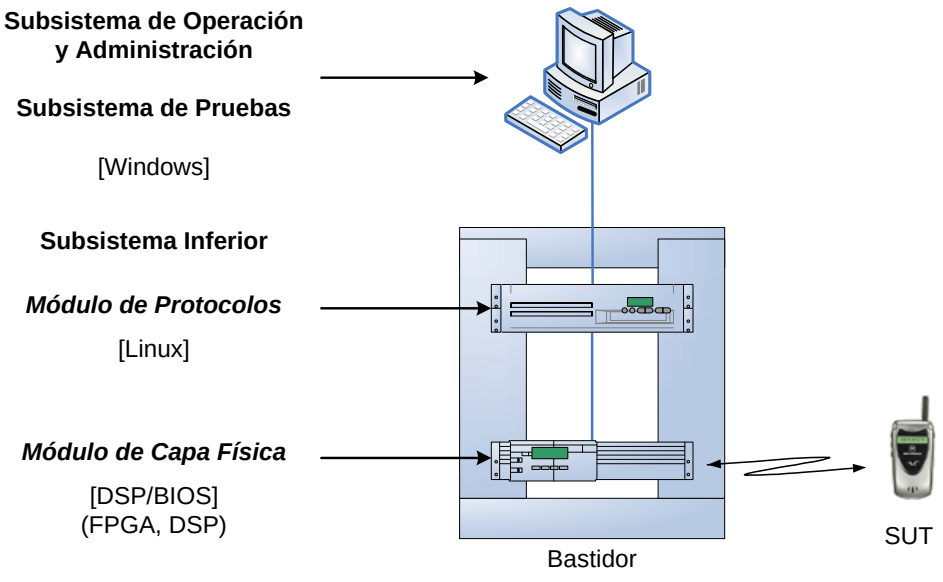


Figura 10.11: Plataforma de Ejecución.

SPICUM servicio de publicaciones

Los componentes del Sistema de Pruebas se ejecutan sobre distintas plataformas (Figura 10.11). El Subsistema de Operación y Administración y el Subsistema de Pruebas se ejecutan ambos sobre una plataforma Windows. El Módulo de Protocolos está formado por un único sistema SDL, que es el encargado de manejar todas las celdas activas. Su plataforma de ejecución es un sistema operativo Linux con núcleo 2.4.20-18.9 [KERN03]. Esta distribución ofrece algunas mejoras que lo acercan a las prestaciones de un sistema operativo de tiempo real ([COOP03], [RIO04])[4]. El Módulo de Capa Física se ha realizado con una tarjeta HERON4-C6201 de Hunt Engineering [HERO00] para cada celda. Esta tarjeta contiene una FPGA (*Field-Programmable Gate Array*) y un DSP C6201 [DSPC6201], con un bus PCI (*Peripheral Component Interconnect*) de hasta 133 MB/s. En las pruebas realizadas se ha medido una velocidad de transferencia alrededor de los 125 MB/s para un *buffer* de 100 KB. El componente Gestión de Entrada/Salida (Capítulo 3, Figura 3.10) es el encargado de enrutar adecuadamente las primitivas entre el Módulo de Protocolos y el Módulo de Capa Física.

10.2.1 Módulo de Protocolos

El Módulo de Protocolos se ha diseñado según las especificaciones de la *Release 99* versión 3.9.0 ([RELE99], [ETSI04]). Su estructura se muestra en la Figura 10.12 ([COLA02]). Contiene tres bloques, dos de los cuales modelan los Niveles RLC y MAC, mientras que el tercero contiene los elementos que son opcionales, los cuales se han modelado como procesos (`BMC` y `RRC_DTX`). Todas las celdas se manejan dentro del mismo sistema SDL; para los canales dedicados se crea una celda con identificador -1. SDL ha sido empleado para el diseño de los protocolos de acceso radio de la tecnología UMTS en diversos sistemas comerciales ([CONT00], [SIPI01], [INTE02], [TAE07]).

De forma global, el comportamiento del Módulo de Protocolos viene gobernado por el reloj del Nivel Físico. La primitiva `PHY_STATUS_IND`, que se recibe en el Nivel MAC, marca el inicio del procesamiento de cada período. El Nivel MAC envía al Nivel Físico los datos a transmitir en el TTI, una primitiva `PHY_DATA_REQ` por cada canal de transporte que puede transmitir en este TTI, y solicita del Nivel RLC los datos a transmitir en el siguiente TTI. La primitiva `MAC_STATUS_IND` indica cuántas PDUs, y de qué tamaño, puede transmitir RLC en el siguiente TTI; se envía una primitiva por cada portadora radio. El Nivel RLC devuelve, en la primitiva `MAC_STATUS_RESP`, las PDUs a enviar. Tras enviar una primitiva para cada portadora radio, el Nivel RLC indica la cantidad de datos pendientes para que el Nivel MAC tenga en cuenta esta información en la planificación del próximo TTI.

El formato de transporte a utilizar se selecciona en función de la capacidad de transmisión solicitada por el Nivel RLC (procedimiento `CalculaTFCSiguiente`). El algoritmo utilizado ([ICUM31]) ordena las portadoras radio por prioridad y elige, para cada canal de transporte, el formato de transporte que mejor se ajusta a la petición realizada. Si la combinación resultante de formatos de transporte no es válida, se va aumentando la capacidad seleccionada en cada canal de transporte hasta encontrar una combinación válida. Si no se encuentra, se escoge la última combinación formada, que será la que permita enviar un mayor porcentaje de las PDUs solicitadas. Una vez

[4] El núcleo del sistema operativo es ahora plenamente desalojable (*preemptable*). Esto permite que al recibir una interrupción se ceda control inmediatamente al proceso que deba atenderla, lo que reduce la latencia notablemente. También se han mejorado los servicios de red con la introducción de NAPI (*New Application Programming Interface*), lo que modifica la forma en que se manejan los paquetes en el núcleo.

SPICUM servicio de publicaciones

seleccionado el formato de transporte de cada canal, se rellenan las primitivas `MAC_STATUS_IND` a enviar en el siguiente TTI.

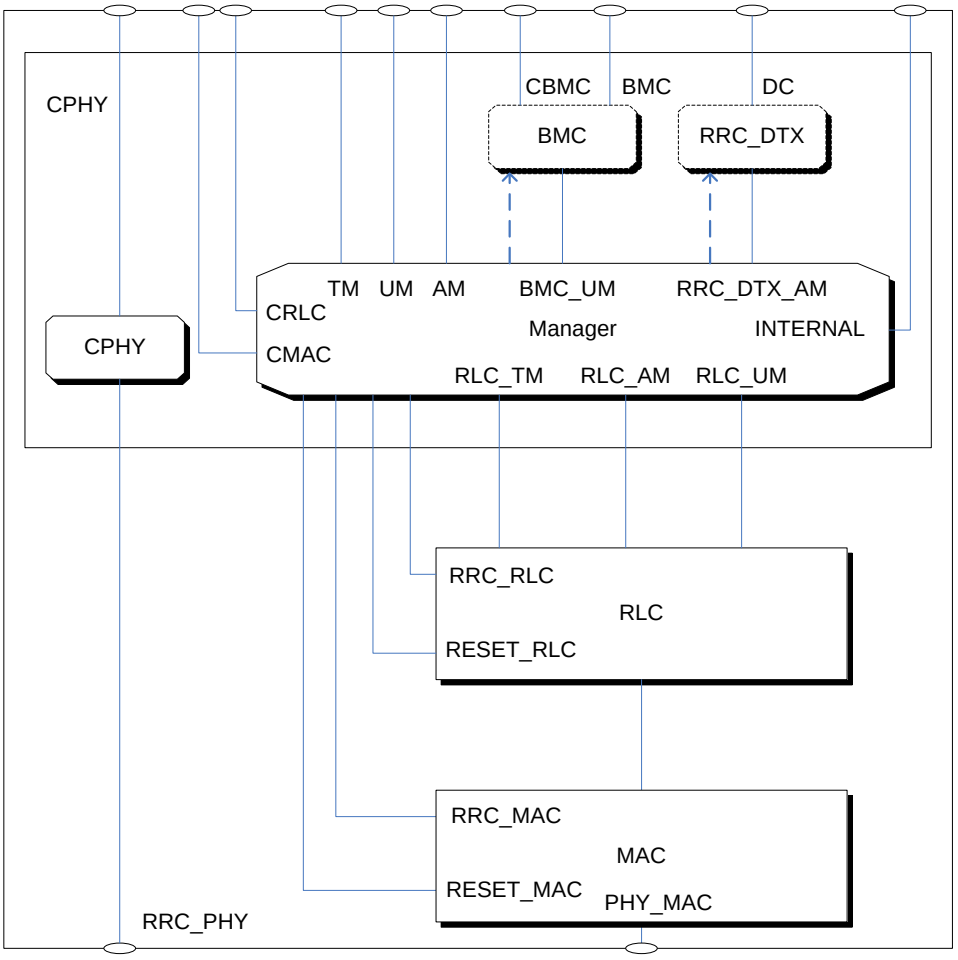


Figura 10.12: Estructura del Módulo de Protocolos en Sistemas de Pruebas para UMTS.

La protección de los mensajes en UMTS se realiza mediante los algoritmos de cifrado (*f8*) e integridad (*f9*) descritos en [3GPP 35.201]; ambos utilizan el algoritmo Kasumi [3GPP 35.202] para el cifrado por bloques. Estos algoritmos se han implementado mediante una librería en C que utiliza el código proporcionado por el 3GPP; su integración se ha verificado con los datos de prueba indicados en [3GPP 35.203]. El uso de estos algoritmos comercialmente requiere una licencia de Mitsubishi.

La protección de la integridad se realiza en los mensajes de señalización transmitidos tras establecimiento de conexión y establecimiento del modo de seguridad. Es un procedimiento que debe aplicarse en el Nivel RRC, pero, dado que las Pruebas no lo implementan, se ha realizado en el multiplexor `MUX_TX` del Nivel RLC. El cifrado se realiza en el Nivel RLC para las portadoras radio que emplean los modos de transferencia UM y AM, y en el Nivel MAC para las que utilizan el modo de transferencia TM. En RLC es el proceso `CTRL` (Sección 10.2.4.2) el que recibe las indicaciones de activación y desactivación del cifrado, aunque la aplicación se realiza en el proceso que maneja la correspondiente portadora radio. En el Nivel MAC el cifrado se realiza en el servicio `MACD` (Sección 10.2.4.3); la cabecera MAC no se cifra.

SPICUM servicio de publicaciones

En la implementación del Módulo de Protocolos se ha mejorado el mecanismo de generación de trazas [ICUM94] respecto al incluido en los Sistemas de Pruebas de Bluetooth. De esta forma, se traza hasta el nivel de símbolo SDL, pudiendo especificar qué tipos de trazas se desea generar y cuáles no.

10.2.2 Generalidades del Diseño

10.2.2.1 ¿Procesos o Servicios?

La decisión sobre la filosofía de diseño utilizada en los Niveles RLC y MAC ha sido tomada tras analizar los beneficios obtenidos por el uso de procesos o servicios[5] [SORE03b]. Los procesos evitan la replicación de las estructuras de datos utilizadas por instancias múltiples del mismo tipo de objeto (por ejemplo, las portadoras radio) y ofrecen el concepto de estado, lo que permite un procesamiento concurrente de distintos flujos. Los servicios, sin embargo, permiten la compartición de variables dentro de todo el bloque (evita rutas de señal y señales para información interna de control)[6] y su código generado es más sencillo que el obtenido a partir de un modelo con procesos.

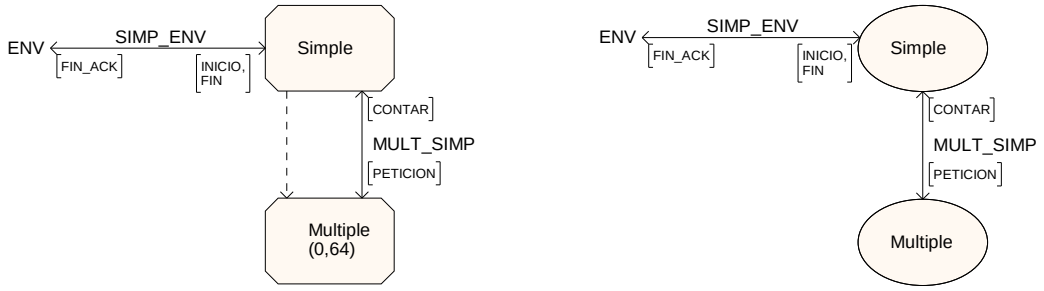


Figura 10.13: Sistemas SDL para la medida de prestaciones de los mecanismos de (a) procesos y (b) servicios.

Se han estudiado dos aspectos relacionados con la velocidad de ambas alternativas: la comunicación de señales y el intercambio de variables. Las medidas ([MORI04], [ICUM17], [ICUM25]) se han realizado con los modelos mostrados en la Figura 10.13, donde un proceso se comunica con 64 instancias de otro y el modelo equivalente empleando servicios. La plataforma utilizada ha sido un equipo Pentium III a 1 GHz con sistema operativo Windows 2000.

Ambos mecanismos ofrecen unas prestaciones similares al medir el número total de señales transmitidas, con un resultado alrededor de las 22000 señales/segundo. Si comparamos la velocidad cuando se intercambian variables (Figura 10.14) de un cierto tamaño (aquí 1340 octetos), los servicios son más rápidos salvo que en el modelo de procesos se utilice código C directamente incrustado en el modelo SDL[7]. El intercambio de variables es instantáneo cuando se usan servicios, ya que las variables se comparten; lo que se está midiendo es la velocidad de asignación.

[5] Los servicios han sido eliminados en la versión SDL-2000, pero las herramientas de diseño no han incorporado esta versión, por lo que siguen estando disponibles.

[6] Esto permite, por ejemplo, activar una configuración instantáneamente.

[7] Este efecto es más acusado conforme aumenta el tamaño de las variables.

SPICUM servicio de publicaciones

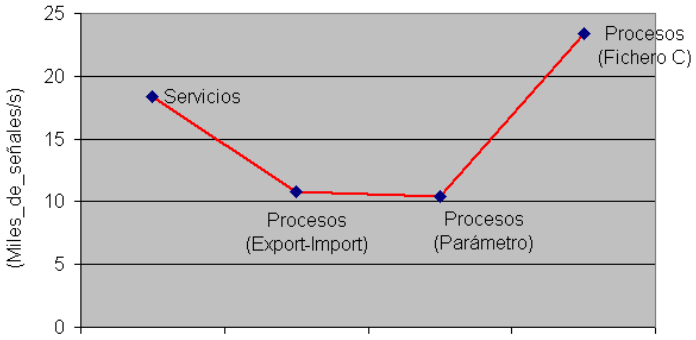


Figura 10.14: Eficiencia del uso de servicios y procesos para el intercambio de variables.

Aunque los servicios son más rápidos en algunos aspectos, la carencia del concepto de estado no los hace adecuados para un procesamiento concurrente de varios flujos, como es el caso de las portadoras radio en el Nivel RLC. En el Nivel MAC, sin embargo, no se requiere un procesamiento concurrente, ya que en cada TTI se procesan secuencialmente cada uno de los canales que deben ser transmitidos. Por ello, a partir de los resultados de este análisis, se decidió modelar el Nivel MAC con servicios y el Nivel RLC con procesos; la compartición de información entre procesos (por ejemplo, los *arrays* que modelan las distintas colas de recepción y transmisión) se ha implementado mediante código C en una librería externa.

10.2.2.2 Orientación a Objetos

Se ha analizado la posibilidad de realizar un diseño que haga un mayor uso de las características de orientación a objetos que ofrece SDL. Según [TAYL92] y [PINS90], los enfoques orientados a objetos pueden reducir el tiempo de desarrollo y el tamaño del código resultante; unas posibles reglas para este tipo de diseños se enumeran en [DERR98].

Dado que el comportamiento del UE (rol de IUT) y la UTRAN (rol de Sistema de Pruebas) es similar, se ha evaluado la posibilidad de implementar tanto el Módulo de Protocolos como la IUT, definiendo el comportamiento común a ambas en un tipo base. La implementación final de cada entidad se realiza en un tipo que hereda de este tipo base; el mecanismo de redefinición permite modificar el comportamiento según la entidad a que corresponda. Se ha estimado que aproximadamente el 50% de los niveles MAC y RLC se podría modelar en los tipos base; este porcentaje disminuye al 10% en el caso del Nivel RRC. La diferencia de porcentaje se debe a las características de cada nivel, ya que la gestión de recursos radio es una función unidireccional mientras que la comunicación de datos es más simétrica. Con este enfoque, se han implementado prototipos del Nivel MAC [CONT03b] y del control de la llamada del Nivel RRC [ALAR01].

Como conclusión se ha estimado que el uso de estas características no proporciona beneficios para el diseño del Módulo de Protocolos de un Sistema de Pruebas, ya que en este caso la complejidad adicional que introducen en el diseño contrarresta las ventajas que ofrecen.

No obstante, las características de orientación a objetos del lenguaje sí se han utilizado como mecanismo para la organización del diseño, mediante la definición de tipos de bloques y procesos que son instanciados posteriormente [COLA03].

SPICUM servicio de publicaciones

10.2.3 Interfaces

Las interfaces externas de los Niveles del Módulo de Protocolos son las especificadas en las normas [3GPP 25.322] (RLC), [3GPP 25.321] (MAC) y [3GPP 25.302] (PHY), salvo la interfaz de control del Nivel MAC (CMAC), que se obtiene de los Juegos de Pruebas. El canal `INTERNO` ofrece una interfaz propia que permite seleccionar diversas configuraciones en el Módulo de Protocolos, no sólo las requeridas por los Juegos de Pruebas, sino también configuraciones propias para realizar pruebas internas [ICUM47]. Cada interfaz se ha declarado en un paquete independiente.

10.2.3.1 Interfaz con el Subsistema de Pruebas

La interfaz con el Subsistema de Pruebas emplea 10 *sockets* que son configurados a través de un fichero (`sap.dat`), que indica el puerto a emplear para cada uno. Las primitivas de la interfaz entre el Subsistema de Pruebas y el Subsistema Inferior están en su mayoría declaradas en notación ASN.1 dentro del Juego de Pruebas [3GPP 34.123-3]. Por ello, se ha utilizado la sintaxis de transferencia utilizada PER. Antecediendo a la primitiva codificada se incluye una cabecera que indica la longitud total (cabecera + primitiva), la celda a la que va dirigida y el tipo de primitiva. El componente Gestión de Entrada/Salida del Módulo Adaptador de los Protocolos del Subsistema Inferior se ha generado automáticamente mediante la herramienta *GenInt*.

Los Juegos de Pruebas se han diseñado de forma que la codificación extremo-extremo de las PDUs de los Niveles RRC, BMC y NAS debe realizarse en el Subsistema Inferior, al tiempo que se le añade la protección de integridad. Por este motivo, las PDUs transportadas por las primitivas `RRC_DataReq`, `RRC_DataInd` y `BMC_DataReq` son codificadas en la sintaxis de transferencia ASCII, ya que deben ser descodificadas en el Subsistema Inferior y éste no dispondría de la información suficiente para ello si se empleara la codificación PER. Una vez codificada la PDU, la primitiva se codifica normalmente en PER.

Por otro lado, los Juegos de Pruebas utilizan las primitivas `RLC_TR_TestDataReq`, `RLC_UM_TestDataReq` y `RLC_AM_TestDataReq` para enviar PDUs de los Niveles MAC y RLC. Estas primitivas se codifican en PER, pero la codificación extremo-extremo de las PDUs se realiza en el Subsistema Inferior. En recepción hay que decidir si la PDU recibida corresponde a una primitiva de datos o a una primitiva `TestData`. Se escoge una u otra alternativa en función de la portadora radio en la que se reciba. Se decodifica como una primitivas `TestData` cuando se recibe en una portadora radio utilizada específicamente para estas primitivas de prueba[8] o si previamente se transmitió una primitiva `TestData` por dicha portadora radio.

10.2.3.2 Interfaz con el Módulo de Capa Física

La comunicación entre el Módulo de Protocolos y el Módulo de Capa Física se realiza a través del Módulo Adaptador de los Protocolos, a diferencia de como se ha realizado en los Sistemas de Pruebas para DECT y Bluetooth. Se ha modificado el componente Gestión de Entrada/Salida para incluir el manejo de esta interfaz. Se utiliza la API ofrecida por Hunt Engineeering [HEAP01], que permite una comunicación asíncrona con una unidad mínima de transferencia de 32 bits. También se ha incluido la posibilidad de

[8] Estas portadoras radio son: (RLC-UM): `RB_UM_7_RLC` (-10), `RB_UM_15_RLC` (-11); (RLC-AM): `RB_AM_7_RLC` (-12), `RB_AM_15_RLC` (-13); (MAC)?? `RB_DCCH_FACH_MAC` (-14), `RB_DCCH_DCH_MAC` (-15), `RB_CCCH_FACH_MAC` (-18).

SPICUM servicio de publicaciones

emular la presencia del Módulo de Capa Física para poder ejecutar el Módulo de Protocolos sin este elemento; esta emulación emplea *sockets* para comunicar la UTRAN y el UE. La sincronización entre el Nivel MAC y el Nivel PHY se realiza a través de un *buffer*. Las primitivas que el Nivel MAC solicita transmitir se almacenan en el *buffer* con un TTI de adelanto para soslayar variaciones en el tiempo de respuesta del Nivel MAC que puedan superar el periodo de TTI.

Figura 10.15: Formato de codificación de las primitivas en la interfaz con el Módulo de Capa Física.

La codificación de las primitivas emplea el formato indicado en la Figura 10.15, estando los campos alineados a 32 bits [ICUM41]. La cabecera está formada por una palabra de sincronismo, una marca de tiempo, el tipo de la primitiva, el número de bloques de 32 bits que vienen en los datos y el identificador de la celda a que corresponde la primitiva. Cada campo de la primitiva se codifica con un esquema LV (*Length-Value* – Longitud-Valor). Por ejemplo, un campo de tipo `Bit_String` se codifica de forma que el primer bloque indica el número de bits incluidos, a continuación se incluyen los bits alineándolos a la izquierda y si el número de bits no es múltiplo de 32, se rellena con ceros. Un campo de tipo `SEQUENCE OF` se codifica indicando en el primer bloque el número de bloques de 32 bits que ocupa el total de los elementos; a continuación se codifica cada uno de los elementos. Para los campos opcionales se rellena el primer bloque con la palabra `0xABABABAB` si está presente y con `0x00000000` si no lo está. El procedimiento de lectura consiste en leer primero la cabecera, extraer el número de bloques a leer a continuación, y leer los datos.

10.2.4 Diseño

A continuación se describe el modelado de los bloques que constituyen el Módulo de Protocolos (Figura 10.12).

10.2.4.1 Bloque GESTOR

Este bloque contiene los elementos que son opcionales en los Métodos de Pruebas. Está gobernado por el proceso `Gestor`, que es el responsable de crear los procesos opcionales y de enrutar las señales de/hacia RLC teniendo en cuenta la configuración seleccionada; además, este proceso permite seleccionar diversas configuraciones para pruebas internas.

El proceso `BMC` actúa como un emulador de parte de la funcionalidad del Nivel BMC [3GPP 25.324]; se utiliza para la realización de las Pruebas de los servicios BMC y SMS.

SPICUM servicio de publicaciones

Este emulador utiliza la portadora radio 30 en sentido descendente sobre un canal lógico CTCH. Los datos se envían en instantes determinados por las Pruebas, notificados previamente a UTRAN y UE, y son generados por las Pruebas ya codificados, por lo que el emulador sólo tiene que generar una primitiva de envío en modo no confirmado, `RLC_UM_DATA_REQ`, y confirmar la primitiva de configuración. El responsable de realizar las planificaciones de nivel 1, suficiente si sólo se va a enviar un mensaje de BMC periódicamente, y de nivel 2, cuando hay más de un mensaje BMC, es el Nivel MAC.

El proceso `RRC_DTX` maneja las primitivas `RRC_DATAREQ` y `RRC_DATAIND` utilizadas por la funcionalidad de transferencia directa. Se emplea para las Pruebas de los niveles superiores (NAS y SMS) y utiliza el modo de transferencia AM del Nivel RLC.

El proceso `CPHY` transforma las primitivas de control del Nivel Físico que utilizan los Juegos de Pruebas, definidas en ASN.1, en tipos de datos más sencillos de procesar por el Módulo de Capa Física. Esta conversión se ha modelado dentro del sistema SDL para que pueda emplearse también durante las simulaciones, ya que en este caso no está disponible el Módulo Adaptador de los Protocolos. Se ha creado un procedimiento para cada tipo de primitiva.

10.2.4.2 Nivel RLC

El Nivel RLC [3GPP 25.322] se ha modelado como un tipo de bloque donde para cada portadora radio se crea dinámicamente un proceso (transparente, no confirmado o confirmado) que controle su operación ([COBA02], [ICUM29], [ICUM33], [ICUM34]). La estructura se muestra en la Figura 10.16.

El Nivel RLC está constituido por un proceso de control (`CTRL`), dos multiplexores en las fronteras superior (`MUX_TX`) e inferior (`MUX_RX`), un tipo de proceso para cada tipo de servicio de transferencia (`TM`, `UM` y `AM`), y un proceso para los datos de difusión. Cada portadora radio[9] es procesada por una instancia del correspondiente tipo de servicio. La interfaz con el nivel superior está constituida por tres puntos de acceso, cada uno de los cuales se emplea para un modo de transferencia. Con el nivel inferior la interfaz emplea un único punto de acceso.

El proceso `CTRL` recibe la configuración del Nivel RLC a través del punto de acceso `CRLCSAP` [ICUM28]. Cada vez que recibe una indicación de crear una nueva portadora radio (`CRLC_CONFIG_REQ`), crea una instancia del tipo de servicio correspondiente, almacena su PId e informa a los multiplexores. El multiplexor `MUX_TX` enruta las primitivas de nivel superior a la instancia correcta en base al PId registrado para la portadora radio; el multiplexor `MUX_RX` hace lo propio con las primitivas del nivel inferior. El multiplexor `MUX_TX` también engloba la funcionalidad del Nivel RRC que no implementan los Casos de Prueba, como la codificación de algunos mensajes.

Cuando se recibe una SDU del nivel superior, se introduce en un *buffer* y se comunica su posición al proceso encargado de procesarla. Este *buffer* es de tipo `Bit_String` para el modo transparente y `Octet_String` para los otros modos. Al estar modelados los modos de transferencia como procesos, para manejar múltiples celdas no es necesario replicar las estructuras de datos necesarias para gestionar una celda, ya que las variables son locales a cada instancia de los tipos de proceso; sólo se requiere un *array* con tantas posiciones como celdas para almacenar las portadoras radio activas en cada una de ellas.

[9] Se pueden tener hasta 64 portadoras radio, que se nombran a través del identificador `RB_Identity` (entero de rango [–31..32]).

SPICUM servicio de publicaciones

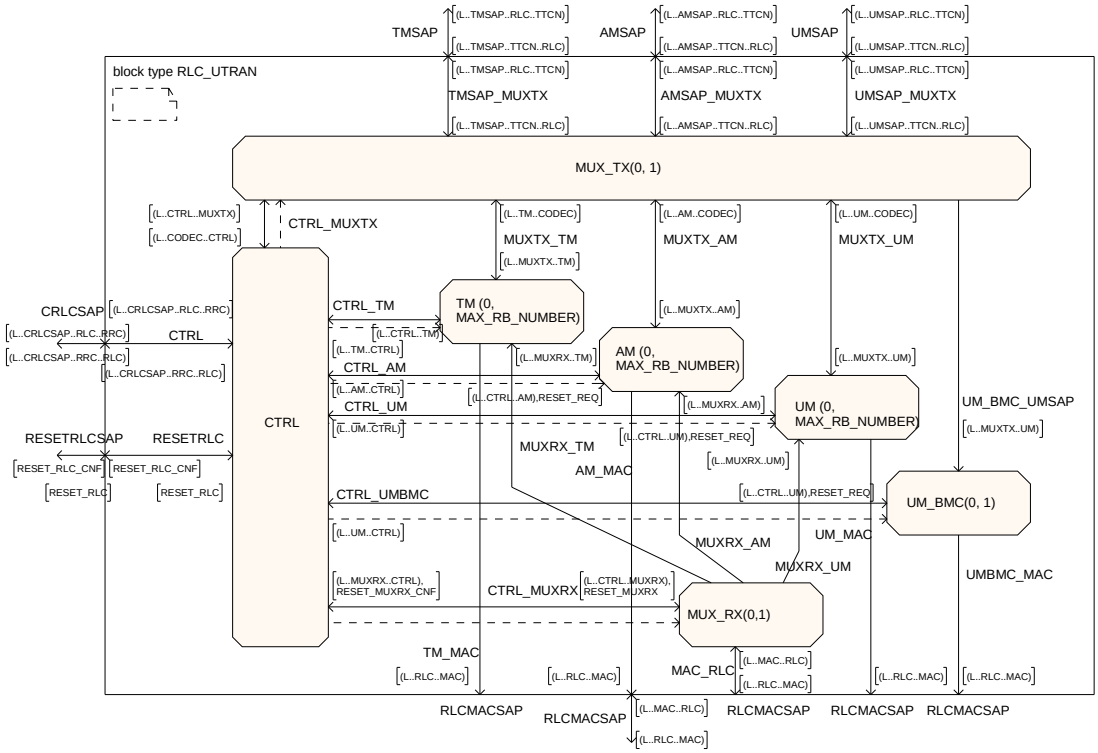


Figura 10.16: Estructura del Nivel RLC.

Cuando el Nivel MAC señaliza el inicio de un nuevo TTI, se construyen las PDUs a enviar, teniendo en cuenta que si la SDU es más grande que la capacidad disponible hay que segmentarla; esta segmentación se realiza en bloques del mismo tamaño, indicado éste por el Nivel MAC. Si cabe más de una SDU en los modos de transferencia UM y AM, y está permitida esta opción, se incluye un campo en la cabecera que indica la longitud de cada una; en el modo de transferencia TM se concatenan sin indicación alguna. Si se emplea el modo de transferencia AM, las SDUs de control tienen prioridad sobre los datos a retransmitir y estos sobre los demás datos. En este modo de transferencia se utilizan también *buffers* adicionales para indicar las PDUs a retransmitir y las pendientes de confirmación.

Antes de construir las PDUs a enviar se decide si hay PDUs que deban ser descartadas. Hay distintos tipos de descarte. Por ejemplo, el modo TM descarta al recibir nuevas SDUs o al haberse sobrepasado un límite temporal, el modo UM incluye el descarte si el *buffer* está lleno y el modo AM también descarta si se supera el número máximo de retransmisiones. En recepción, las PDUs recibidas se reensamblan en una o más SDUs y, una vez completada ésta, se entregan inmediatamente al nivel superior[10]. El formato de las PDUs de datos para los modos UM y AM se muestra en la Figura 10.17.

El proceso `UM_BMC` maneja la portadora radio 30 en sentido descendente, que es la utilizada por el emulador de BMC; en esencia es un modo de transferencia UM simplificado. Cuando recibe datos del nivel superior (BMC) los procesa y los entrega al Nivel MAC; no tiene en consideración los límites temporales de TTI.

[10] La norma indica que la ventana de recepción puede ser de hasta 4096 PDUs, pero en las pruebas no se configura a más de 8, por lo que se ha escogido utilizar un tamaño máximo de 32.

SPICUM servicio de publicaciones

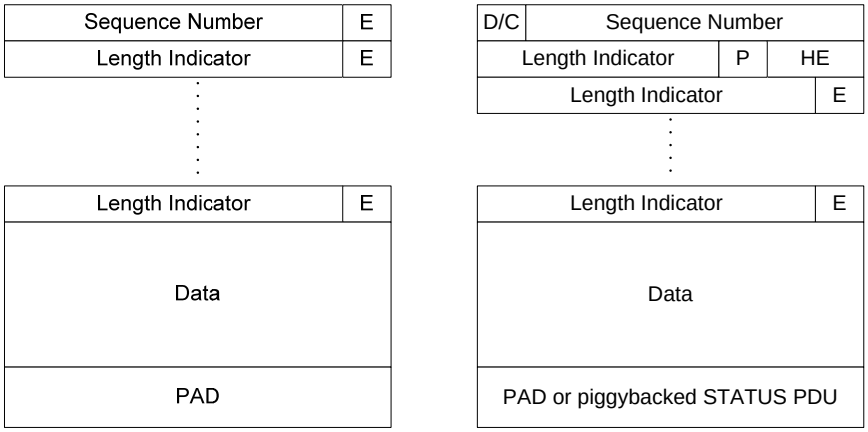


Figura 10.17: Formato de las PDUs de datos de los modos (a) UM y (b) AM.

Los *buffers* utilizados en el Nivel RLC se manejan a través de una librería externa modelada en C, ya que ofrece una mayor eficiencia en la ejecución. Los *buffers* se han implementado como colas circulares de 256 posiciones. La Figura 10.18-a lista los procedimientos empleados para los dos tipos de elementos manejados: `Bit_String` y `Octet_String`. La Figura 10.18-b muestra cómo se declara un procedimiento externo para su uso dentro del modelo SDL y la forma de invocarlo.

Octet_String	Bit_String
Init_Buffers	
Add_Sdu	Add_BitSdu
Get_Sdu	Get_BitSdu
Discard_Sdu	Discard_BitSdu
Length_Sdu	Length_BitSdu

(a)

```
/* Declaración */

procedure Add_Sdu;
  fpar in Sdu Octet_String,
  in/out position Integer;
returns Integer;
external;

/* Invocación */

res:=Call Add_Sdu (sdu, index);
```

(b)

Figura 10.18: (a) Lista de funciones externas para manejo de los buffers RLC y (b) Declaración y uso en SDL.

10.2.4.3 Nivel MAC

El Nivel MAC [3GPP 25.321] ha sido modelado como un tipo de proceso cuyo comportamiento está descrito mediante servicios ([SORE03b], [ICUM32], [CONT03b]) La estructura se muestra en la Figura 10.19. Contiene cinco servicios que se encargan de la configuración y el control de este Nivel (`CTRL`), multiplexar las fronteras superior (`MUX_TX`) e inferior (`MUX_RX`), y procesar los canales de transporte dedicados (`MACD`) y compartidos (`MACC_SH`). A diferencia del Nivel RLC, al estar modelado como servicios, para poder manejar varias celdas cada variable se declara como un *array* con tantas posiciones como celdas. Un ejemplo del uso de estas declaraciones es

```
bufferConfigDL(cell_id)(stringConfigDL(cell_id)(i))!numLogCH
```

SPICUM servicio de publicaciones

donde el segundo índice del *array* externo se obtiene indexando otro *array* también con el identificador de la celda.

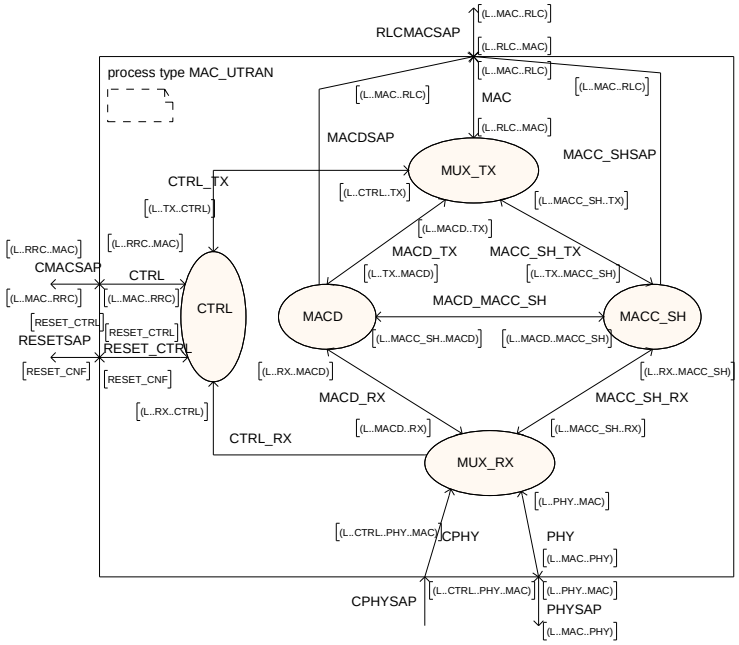


Figura 10.19: Estructura del Nivel MAC.

El servicio `CTRL` es el encargado de seleccionar la combinación de formatos de transporte a emplear en cada TTI; la configuración de las posibles combinaciones de formatos de transporte se recibe a través de la primitiva `CMAC_CONFIG_REQ`. Los servicios `MACD` y `MACC_SH` son responsables de la multiplexación de canales y el procesamiento de la cabecera. Las cabeceras se emplean en los canales de transporte compartidos y en los canales de transporte dedicados donde se multiplexan varios canales lógicos. Su tamaño depende tanto del tipo canal de transporte como del tipo de canal lógico mapeado en él (Tabla 10.6). Adicionalmente, el servicio `MACC_SH` maneja el canal de difusión y el servicio `MACD` realiza el cifrado[11] si está activado. Los datos recibidos se entregan en cuanto se reciben; si se reciben varias PDUs por la misma portadora radio, se indican con una sola primitiva.

El multiplexor de recepción (`MUX_RX`) decide si entregar los datos recibidos a `MACD` (canales DCH) o a `MACC_SH` (resto) a través del identificador del canal de transporte utilizado. En sentido descendente, a partir de la planificación recibida del Nivel RRC y con la información de los campos MIB (*Master Information Block*) y SIB (*System Information Block*), construye la información a enviar por el canal de difusión en cada TTI. La planificación habitual que se utiliza en las Pruebas se muestra en la Tabla 10.7.

La correspondencia entre canales lógicos y canales de transporte se almacena en un *array* [ICUM30], donde cada uno de los 32 elementos representa un canal de transporte e indica los canales lógicos que se multiplexan (hasta 15) sobre él. Para realizar consultas sobre este *array* de forma rápida, hay una variable asociada de tipo `String`, que almacena los identificadores de los canales de transporte activos. Los conjuntos de

[11] El cifrado se realiza solamente sobre la PDU del Nivel RLC, no sobre la cabecera del Nivel MAC.

SPICUM servicio de publicaciones

formatos de transporte se almacenan en *arrays* de longitud 32, uno para cada sentido (`tfssUL`, `tfssDL`), así como las combinaciones de formatos de transporte, de las cuales puede haber configuradas hasta 1024.

Tabla 10.6: Tamaño de las cabeceras MAC en función de los tipos de canal.

Canal de Transporte	Canal lógico	Tamaño (octetos)
DCH (numLogCH=1)	-	0
DCH (numLogCH>1)	-	4
BCH	-	0
PCH	-	0
FACH	BCCH	2
FACH	CCCH	8
FACH	CTCH	8
FACH	DCCH	40
FACH	DTCH	40
RACH	CCCH	2
RACH	DCCH	24
RACH	DTCH	24
DSCH (numLogCH=1)	DTCH	18
DSCH (numLogCH=1)	DCCH	18
DSCH (numLogCH>1)	DTCH	22
DSCH (numLogCH=1)	DCCH	22

Tabla 10.7. Planificación típica del canal de difusión en las Pruebas.

TTI	*Bloque*	*TTI*	*Bloque*	*TTI*	*Bloque*	*TTI*	*Bloque*
0	MIB	**8**	MIB	**16**	MIB	**24**	MIB
1	SB1	**9**	SB1	**17**	SB1	**25**	SB1
2	SIB7	**10**	SIB3 + SIB7	**18**	SIB18 + SIB7	**26**	SIB4 + SIB7
3	SIB6 (seg1)	**11**	SIB1 + SIB2	**19**	SIB5 (seg1)	**27**	No usada
4	MIB	**12**	MIB	**20**	MIB	**28**	MIB
5	SIB6 (seg2[12])	**13**	SIB12 (seg1)	**21**	SIB5 (seg2[12])	**29**	SIB11 (seg1)
6	SIB6 (seg3[12])	**14**	SIB12 (seg2[12])	**22**	SIB5 (seg3[12])	**30**	SIB11 (seg2[12])
7	SIB6 (seg4[12])	**15**	SIB12 (seg3[12])	**23**	SIB5 (seg4[12])	**31**	SIB11 (seg3[12])

10.2.5 Pruebas

Las Pruebas de Módulo se han realizado sobre el sistema SDL mostrado en la Figura 10.20. El lado izquierdo de la figura muestra la torre de protocolos correspondiente a la UTRAN y el lado derecho la correspondiente al UE. Los Módulos de Capa Física se emulan en los bloques `PHY_UTRAN` y `PHY_UE`[13]. Al conjunto formado por el Módulo de Protocolos y los emuladores de Nivel Físico se le ha denominado Sistema Virtual de Pruebas. Este sistema permite verificar el funcionamiento de los protocolos de Nivel 2 y superior sin la necesidad de un Módulo de Capa Física real.

El emulador del UE utilizado en las Pruebas de Módulo se obtiene básicamente invirtiendo los canales unidireccionales (ascendente ↔ descendente) y adaptando

[12] Si es necesario. En caso contrario se envía '*MsgNoSegment*'.

[13] La comunicación entre cada entidad se realiza mediante *sockets* configurables a través de un fichero.

SPICUM servicio de publicaciones

adecuadamente la máquina de estados. Incluye la funcionalidad necesaria para la realización de las pruebas y dispone de diversas configuraciones seleccionables. Se han empleado dos tipos de pruebas: propias y Juegos de Pruebas. Las pruebas propias se listan en la Tabla 10.8 [ICUM49]. Una vez superadas estas pruebas, se han ejecutado los Juegos de Pruebas oficiales. El UE se configura a través de un emulador de RRC; desde el exterior del sistema se le indica la configuración que debe fijar[14]. Las Pruebas de Subsistema (integración con el Módulo de Capa Física) y de Sistema las ha realizado AT4 wireless.

Tabla 10.8: Lista de Pruebas de Módulo propias realizadas.

Grupo	Nombre	Descripción
Grupo A: Una portadora radio	RB[15] modo TM	Probar una única portadora radio TM en ambos sentidos (RLC). Probar un canal lógico DCCH mapeado en un canal de transporte DCH (MAC).
	Segmentación/ no segmentación en modo TM	Probar una única portadora radio TM que segmenta en ambos sentidos y que no segmenta en la dirección ascendente.
	RB modo UM con LI[16] de 7 bits	Probar una única portadora radio UM en ambos sentidos (RLC) con *Length Indicators* de 7 bits. Probar un canal lógico CCCH mapeado en un canal de transporte FACH y CCCH mapeado en un RACH (MAC).
	RB modo UM con LI de 15 bits	Probar una dirección de una portadora radio UM con *Length Indicators* de 15 bits (RLC).
	RB combinada TM Ascendente/UM Descendente	Probar una portadora radio TM en la dirección ascendente y UM en la dirección descendente (RLC). Probar un canal lógico CCCH mapeado en un canal de transporte FACH y CCCH mapeado en un RACH (MAC).
	RB combinada TM Ascendente/UM Descendente + Cifrado	Probar el cifrado en RLC (modo UM) y en MAC (modo TM en canal DCH).
	RB modo AM con LI de 7 bits	Probar una única portadora radio AM en ambos sentidos (RLC) con *Length Indicators* de 15 bits.
	RB modo AM con LI de 15 bits	Probar una única portadora radio AM en ambos sentidos (RLC) con *Length Indicators* de 15 bits.
	RB modo AM con distintas combinaciones de canal lógico y canal de transporte	Probar distintas cabeceras MAC.
	RB modo AM en canal de difusión	Probar el canal de difusión.
	Elección de formato de transporte	Probar la correcta elección del formato de transporte en una única portadora radio.
Grupo B: Varias portadoras radio	RBs de señalización (RBs 1, 2, 3, 4) + Protección de integridad	Probar la protección de integridad.
	Canal señalización 3,4Kbps	Probar la configuración y correcto funcionamiento de un canal de señalización de velocidad 3,4 Kbps tal y como se configura en los Juegos de Pruebas.
	Canal de voz 12,2 Kbps + 3,4 Kbps de señalización	Probar la configuración y correcto funcionamiento de un canal de voz de 12,2 Kbps más un canal de señalización de velocidad 3,4 Kbps tal y como se configura en los Juegos de Pruebas.

[14] Los mensajes del UE utilizan el conjunto de comandos AT, para lo cual se ha implementado el codificador correspondiente en la aplicación de control.

[15] La abreviatura RB denota a una portadora radio (RB – *Radio Bearer*)

SPICUM servicio de publicaciones

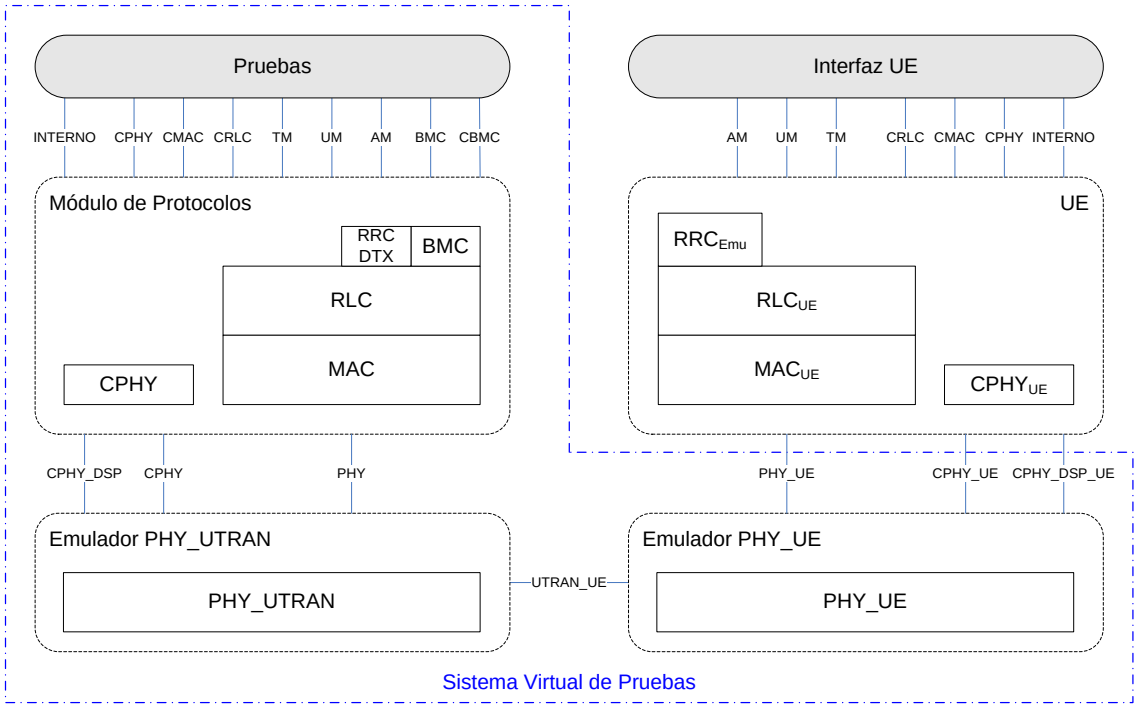


Figura 10.20: Arquitectura del sistema empleado para las Pruebas de Módulo.

Las prestaciones del Módulo de Protocolos implementado se han evaluado sobre los sistemas operativos Linux 2.4.20, en un equipo Pentium III a 533 MHz, y Windows 2000, en un equipo Pentium IV a 1 GHz [SORE03b]. Las velocidades de transferencia consideradas han sido las empleadas por las portadoras configuradas por las Pruebas: 12,2 kbps, 38,4 kbps y 384 kbps. En la Figura 10.21 se muestran los resultados obtenidos para la velocidad máxima considerada. En este caso, el sistema operativo Windows no alcanza la velocidad configurada, produciéndose un deterioro de la tasa de bit conforme se incrementa el intervalo medido; por su parte, Linux sí alcanza la tasa deseada de 384 kbps. No se han evaluado otra vez las prestaciones en el sistema operativo utilizado finalmente; al poseer características más próximas a un sistema de tiempo real, las prestaciones serán mejores que las medidas realizadas anteriormente.

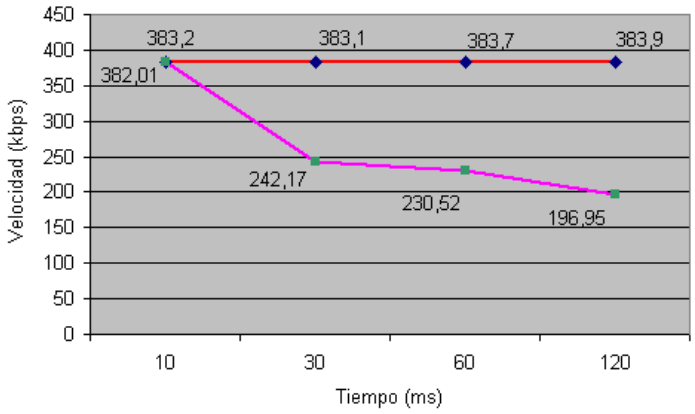


Figura 10.21: Velocidades alcanzadas con una configuración de portadora de 384 kbps en Windows y Linux.

[16] *Length indicator*. Es un campo de la cabecera RLC.

SPICUM servicio de publicaciones

10.3 Sistema de Pruebas de Interoperatividad

Siguiendo la línea iniciada en Bluetooth, se ha construido un Sistema de Pruebas de interoperatividad para el Nivel RRC de UMTS. Se han modelado los escenarios específicos que requieren estas pruebas para situar al Equipo Bajo Prueba en el estado adecuado, de forma que simule el comportamiento de un Nodo B. El proceso de diseño se ha formalizado parcialmente, haciendo uso del entorno de desarrollo e integrando herramientas para la generación automática de código a partir de los Juegos de Pruebas de conformidad.

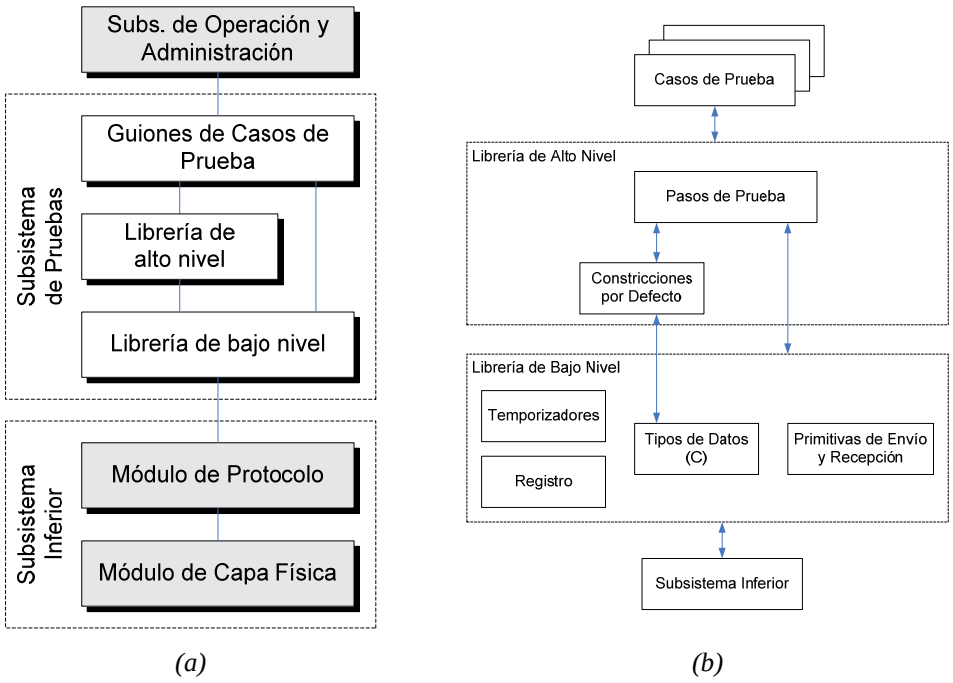


Figura 10.22: (a) Arquitectura del Sistema de Pruebas de interoperatividad y (b) Detalle de las librerías.

El diseño del Sistema de Pruebas puede ser personalizado por el usuario, ya que le permite definir y construir sus propios escenarios específicos de prueba, haciendo uso de las interfaces de programación definidas. Este Sistema de Pruebas se ha utilizado también como Unidad de Señalización del Sistema de Pruebas radio de UMTS.

La arquitectura del Sistema de Pruebas se muestra en la Figura 10.22. La funcionalidad de los Niveles inferiores al Nivel RRC está proporcionada por el Subsistema Inferior descrito en la Sección anterior. El Subsistema de Pruebas incluye las secuencias de prueba que el usuario pueda definir. El usuario puede emplear dos librerías, implementadas en C, según la flexibilidad que requiera. La interfaz de bajo nivel ofrece servicios básicos, como son la transmisión y recepción de primitivas, el manejo de tipos de datos, la codificación de la información, la gestión de temporizadores, funciones para el registro de la ejecución, asignación de veredictos e, incluso, acceso directo a los servicios en la frontera superior del Subsistema Inferior. Sin embargo, los escenarios que normalmente quiere diseñar un usuario son escenarios de uso habitual con algunas modificaciones. Por ello, se proporciona también una interfaz de programación de alto nivel, que ofrece funciones al modo de los Pasos de Prueba en TTCN; son utilizados para modelar los Casos de Prueba. Por ejemplo, el usuario puede iniciar el envío de la

SPICUM servicio de publicaciones

información de difusión con los valores empleados por defecto en las pruebas (`SendDefaultSysInfo`) o establecer una llamada (`GenericCallSetup`).

Tabla 10.9: Reglas de conversión de las definiciones de tipos ASN.1 en C.

<table>
<tr><th>ASN.1</th><th>C</th><th>ASN.1</th><th>C</th></tr>
<tr><td>INTEGER
BOOLEAN
ENUMERATED</td><td>typedef int Valor;</td><td>OCTET_STRING
BIT_STRING
HEXSTRING
IA5String</td><td>typedef char * String;</td></tr>
<tr><td>SEQUENCE
{
  field1 Field1,
  field2 Field2,
  ...
  fieldN FieldN
  OPTIONAL
}</td><td>typedef struct
Sequence {

  struct {
  int any;
  Field1 field;
  } f_field1;

  ...

  struct {
  int any;
  int present;
  FieldN field;
  } f_fieldN;

} Sequence;</td><td>CHOICE
{
  field1 Field1,
  field2 Field2,
  ...
  fieldN FieldN
}</td><td>typedef struct Choice {
  int id;
  union {
  struct {
  int any;
  Field1 field;
  } f_field1;

  ...

  struct {
  int any;
  FieldN field;
  } f_fieldN;

  };
} Choice;</td></tr>
<tr><td>SEQUENCE OF Element</td><td colspan="3">typedef struct SequenceOf {
  Element field;
  SequenceOf *next;
} SequenceOf;</td></tr>
</table>

La librería de bajo nivel reutiliza en lo posible el código generado por el entorno de desarrollo y las herramientas existentes. Por ello, incorpora tanto el Motor TTCN del Subsistema de Pruebas como los Módulos Adaptador y de Gestión de las Pruebas, así como parte del código generado a partir de los Juegos de Pruebas. El manejo de los tipos de datos, tanto su construcción como su comparación con otro valor, se ha encapsulado en funciones que ofrecen una interfaz más cómoda para el desarrollador, pero internamente hacen uso de la interfaz GCI. También se generan funciones de envío y recepción, una para cada primitiva, que utilizan internamente, como parámetros, los tipos de datos de la interfaz de bajo nivel.

Tabla 10.10: Ejemplo de código para acceder a los campos de un tipo estructurado.

Código del usuario	Código tras el preprocesado

Sequence seq;
Field1 elem;
 	
seq.field1 = elem; /*or */
seq.field1_any = ANY_VALUE;
 	
/*field1 is present*/
seq.field1_present = 1;
	#define field1 f_field1.field
	#define field1_any f_field1.any
	#define field1_present f_field1.present
	
	seq.f_field1.field = elem; /*or */
	seq.f_field1.any = ANY_VALUE;
	
	/*field1 is present*/
	seq.f_field1.present = 1;

SPICUM servicio de publicaciones

Las funciones de manejo de los tipos de datos, para convertir de tipo de usuario a tipo interno (prefijo `Get`) y viceversa (prefijo `Mk`), se generan automáticamente a partir de la lectura de las declaraciones de los tipos de datos en los Juegos de Pruebas de conformidad[17]. La conversión de las declaraciones de los tipos ASN.1 en tipos C se ha realizado siguiendo las reglas indicadas en la Tabla 10.9. Los campos opcionales se han modelado con una estructura con un campo que indica si está presente. En el caso de un tipo `CHOICE`, el primer campo indica cuál de todas las opciones está presente. Mediante macros se oculta esta estructura interna de los tipos de datos, de forma que el usuario escribe un código similar al mostrado en la columna izquierda de la Tabla 10.10; tras el preprocesado el código se expande como se muestra en la columna derecha.

```
/*Init SU*/
if ((status = InitSU(135000, u_ms))!=MATCH) return status;

/*Set RF Test Conditions*/
SetFrequencyInfo(c_cellA_id, 10563, NULL);

/*Configure SU*/
if ((status = CreateCellDCH_Modified(c_cellA_id))!=MATCH) return status;

/*Send Default SysInfo*/
if ((status = SendDefaultSysInfo(c_cellA_id))!=MATCH) return status;

/*Idle Update*/
if ((status = IdleUpdated(c_cellA_id))!=MATCH) return status;

/*Call Setup*/
if ((status = GenericCallSetup(c_cellA_id, e_MEAS_CHANNEL_12_2kbps))!=MATCH)
  return status;

/*Release UE Connection*/
if ((status = RRC_ConnectionRelease(c_cellA_id))!=MATCH) return status;

/*Release SU*/
if ((status = Release(c_cellA_id))!=MATCH) return status;
```

Figura 10.23: Ejemplo de uso de la librería de alto nivel para el establecimiento y liberación de una llamada.

La librería de alto nivel se ha construido manualmente sobre los servicios ofrecidos por la librería de bajo nivel. Proporciona unos ciento cuarenta Pasos de Prueba y otras funciones que asignan los valores por defecto a variables de cerca de cuatrocientos tipos diferentes, equivalentes a las restricciones (*constraints*) en TTCN. Esta librería permite a los usuarios más experimentados realizar cambios en función de sus requisitos. Un ejemplo de cómo se establece y libera una llamada haciendo uso de las funciones de esta librería se muestra en la Figura 10.23.

10.4 Sistemas Comerciales

En la tecnología UMTS ha habido una oferta más amplia de Sistemas de Pruebas de conformidad tanto de protocolos como de radio que en tecnologías anteriores. Su comercialización requiere la obtención previa de la certificación GCF (*Global Certification Forum*). Las Pruebas de protocolos han ido aumentando su complejidad con cada nueva generación de tecnologías, por ello, una característica común a todos los

[17] Esta herramienta está basada en la herramienta *GenInt*., pero genera una salida diferente y ha sido extendida para manejar tipos ASN.1.

SPICUM servicio de publicaciones

Sistemas de Pruebas es que permiten fijar secuencias de pruebas que se ejecuten automáticamente, ya que la realización de todas las Pruebas puede durar varios días.

Los productos más significativos han venido de la mano de Anritsu, Anite, AT4 wireless y Rohde & Schwarz. Además, hay una amplia variedad de otros Sistemas de Pruebas diseñados para entornos de producción o verificación de desarrollos. Aunque se comentan aquí las características principales de los Sistemas de Pruebas para UMTS, en general estos se ofertan como una variante dentro de una línea de productos para distintas generaciones (2G/2.5G/3G/3.5G) de comunicaciones móviles, como GSM, GPRS, EDGE, HSDPA, WiMAX y LTE.

Anritsu ha puesto en el mercado una extensa gama de productos [ANRIUM]. Su base es la unidad de señalización MD8480 en sus variantes B y C. Este equipo permite la realización de pruebas de protocolos, de modulaciones radio y de aplicaciones de voz, vídeo y datos. El modelo MD8480C incluye una interfaz de programación en C que permite al usuario crear nuevos escenarios de prueba. Sobre esta unidad, Anritsu ofrece un Sistema de Pruebas de conformidad de protocolos, MX785201A (PTS), para los Niveles 2 y 3, con capacidad para realizar pruebas de interoperatividad (Figura 10.24). Además, este suministrador ofreció durante un tiempo un entorno de pruebas virtual, MX785101A (VST), para comprobar el funcionamiento de protocolos de los Niveles 2 y 3 empleando una emulación del Nivel Físico.

Figura 10.24: Sistema de Pruebas para protocolos de UMTS comercializado por Anritsu.

Anite utiliza su plataforma SAT para ofrece Sistemas de Pruebas tanto para protocolos como para radio [ANITUM]. Esta plataforma dispone de hasta 8 transceptores y permite diseñar Casos de Prueba personalizados en TTCN o a través de una API en C. Su producto SAT-H proporciona el software necesario para ejecutar los Juegos de Pruebas del 3GPP en esta plataforma.

AT4 wireless ha ofrecido Sistemas de Pruebas de conformidad tanto para protocolos como para RF dentro de su familia de productos MINT (*Mobile Communications INtegrated Tester*) [MINT]. Como ya se ha indicado en secciones anteriores, los protocolos de Nivel 2 de estos productos han sido uno de los resultados de nuestra colaboración con esta compañía. El Subsistema de Operación y Administración incluye una funcionalidad equivalente a la proporcionada en los Sistemas de Pruebas de Bluetooth: análisis de los mensajes y visualización gráfica de las interacciones con el Sistema Bajo Prueba, realización automática de secuencias de Pruebas, generación automática de informes de certificación, etc. Además, ofrece la posibilidad de utilizarlo como un Sistema de Pruebas de interoperatividad para el Nivel RRC y superiores a través de una API en C. El Sistema permite hasta 8 celdas independientes.

SPICUM servicio de publicaciones

Rohde & Schwarz dispone de una gama de productos basados en la plataforma CRTU, enfocada tanto para pruebas de conformidad como para pruebas de desarrollo [ROHDUM]. Es un producto que ha evolucionado desde las pruebas de GSM. El Sistema de Pruebas para protocolos incluye un Subsistema de Operación y Administración similar al de otros suministradores y un editor de TTCN, la herramienta Leonardo, para visualizar los Casos de Prueba. En su Sistema de Pruebas, Rohde & Schwarz incorpora un enfoque similar al que se ha empleado para el Sistema de Pruebas de interoperatividad, con dos APIs en C que denomina LLAPI (*Low Level API*) y MLAPI (*Medium Level API*). Para facilitar el diseño de nuevos Casos de Prueba, junto con la API de nivel medio ofrece el código fuente y diversos escenarios ya modelados.

10.5 Conclusiones

La participación en el diseño de los Sistemas de Pruebas de conformidad para UMTS ha sido un paso más que ha permitido evolucionar la Metodología de Diseño y las herramientas asociadas a la misma. Las Especificaciones de Prueba definen una única arquitectura de pruebas, de forma que los Puntos de Observación y Control son los mismos para las Pruebas de cualquier nivel de protocolos. Por ello, la estructura del Módulo de Protocolos permite seleccionar dinámicamente su configuración interna en función de las Pruebas a realizar. Por otra parte, los Juegos de Pruebas utilizan más ampliamente la notación ASN.1 para la definición de las primitivas y mensajes los Juegos de Pruebas en detrimento de las construcciones nativas de TTCN. Esto ha llevado a extender los generadores automáticos de interfaces de que ya se disponía incorporando la sintaxis de transferencia PER en la interfaz entre el Subsistema de Pruebas y el Subsistema Inferior.

La mayor velocidad de UMTS en el acceso radio impone requisitos más estrictos en la interacción entre el Módulo de Protocolos y el Módulo de Capa Física. El código generado a partir de SDL realiza con holgura las operaciones necesarias cada período de trama, pero exige que la plataforma atienda a este proceso en momentos específicos, algo que no es posible conseguir con sistemas operativos como Windows. Por ello, se ha empleado una versión del sistema operativo Linux con características próximas al tiempo real. Es posible manejar hasta ocho Módulos de Capa Física, uno por celda, dentro del mismo Módulo de Protocolos. Para el manejo eficiente de las PDUs en el Nivel RLC se requiere el manejo de diversas colas. El uso del lenguaje SDL para ello es ineficiente, por lo que se ha realizado mediante una librería externa en C cuyas funciones son invocadas desde el modelo SDL.

El Módulo de Protocolos diseñado ha sido utilizado en Sistemas comerciales de Pruebas de conformidad tanto de protocolos como de radio (familia MINT). En este segundo caso forma parte de la Unidad de Señalización necesaria para estas pruebas. Además, también se ha empleado en un Sistema de Pruebas de interoperatividad, personalizable a las necesidades de los usuarios mediante la API proporcionada. Se ha avanzado en la formalización del proceso de diseño de este tipo de Sistemas de Pruebas, para lo cual se han tomado como base los Métodos y Juegos de Pruebas de conformidad estandarizados.

SPICUM servicio de publicaciones

SPICUM
servicio de publicaciones

Capítulo

11

"¿Sabes? He estado pensando: utilizar la radio en una forma de vida de esa clase resultaría natural, por así decir, no un producto de la tecnología..."

Nigel Wamsley, A Través del Mar de Soles

CAPÍTULO 11: SISTEMAS DE PRUEBAS RADIO

Este capítulo cierra la sección de Implementaciones. Mientras que los capítulos anteriores han estado dedicados a los Sistemas de Pruebas de protocolos, el objeto de este capítulo es la aplicación de la Metodología de Diseño descrita en los capítulos 4 y 6 al diseño de Sistemas de Pruebas radio.

Se han construido Sistemas de Pruebas radio para las tecnologías Bluetooth y UMTS ([PONC07a], [PONC07b], [GOME01a]) siguiendo la arquitectura expuesta en el capítulo 6. Se ha validado tanto el uso de la interfaz de primitivas propuesta para el control de la instrumentación como el Módulo de Acceso a la Instrumentación implementado. Como instrumentación se han utilizado equipos de diferentes fabricantes. Al no existir una lista aceptada de Instrumentos Virtuales se han considerado como tales los distintos equipos reales empleados para las Pruebas. Las Pruebas se han realizado en modo conducido.

En primer lugar, se comentan las ideas generales en que se ha basado el diseño. A continuación, se describen los Sistemas de Pruebas radio construidos. Finalmente, se describe en mayor detalle uno de los Casos de Prueba implementados para UMTS.

11.1 Generalidades del Diseño de los Juegos de Pruebas

El modelado de las Pruebas se ha tratado de hacer lo más uniforme y modular posible. Para nombrar los instrumentos y los comandos empleados se han definido constantes; el uso de prefijos diferentes permite diferenciar entre instrumentos (`TSC_id_`), comandos (`TSC_com_`) y parámetros (`TSC_par_`)[1]. Estas constantes y los parámetros de las primitivas de la interfaz de control de la Instrumentación se han declarado de tipo `IA5String`. En el Apéndice L se listan los comandos y parámetros utilizados.

[1] Los mensajes al operador se han almacenado también en constantes con el prefijo `TSC_mmi_`.

SPICUM servicio de publicaciones

Los Parámetros de Pruebas se han obtenido de las Especificaciones de Prueba para cada tecnología ([BTEST RF], [3GPP 34.121]). Para implementar algoritmos de complicado modelado en TTCN, como por ejemplo los de procesado de señales, se han definido funciones TSO; algunos ejemplos son `TSO_Access_Code`, que devuelve la palabra binaria de sincronismo de Bluetooth, o `TSO_Calc_SpecEmissMask`, que determina si se cumple la máscara de emisión espectral establecida. El conjunto de estas funciones se podría generalizar para definir una librería de funciones que fuera empleada en el modelado de Juegos de Pruebas radio.

Los Casos de Prueba radio se pueden ver como una secuencia de interacciones mediante la cual la Prueba solicita de la Instrumentación que establezca unas ciertas condiciones en el medio (en nuestro caso, en una banda de frecuencias de la interfaz aire) para, posteriormente, solicitar de esta misma Instrumentación la realización de un conjunto de medidas y el envío de los resultados a la Prueba, la cual establece el veredicto oportuno.

Test Step Name		`Get_parameter_err_rsp`	
Group		`Err_rsp/`	
Objective			
Default		`Check_T_global(T_trmca01e)`	
Comments			
Nr	**Label**	**Behaviour Description**	**Constraints Ref**
1		`START T_error_com`	
2		`GPIB ? GET_PARAMETER_RSP (TCV_par := GET_PARAMETER_RSP.par_devuelto, TCV_cod := GET_PARAMETER_RSP.cod_error, TCV_cad := GET_PARAMETER_RSP.cad_error)`	`Get_parameter_rsp (?, TSC_no_error, ?)`
3		`CANCEL T_error_com`	
4		`GPIB ? GET_PARAMETER_RSP (TCV_par := GET_PARAMETER_RSP.par_devuelto, TCV_cod := GET_PARAMETER_RSP.cod_error, TCV_cad := GET_PARAMETER_RSP.cad_error)`	`Get_parameter_rsp (?, ?, ?)`
5		`CANCEL T_error_com`	
7		`+ Cerrar_sistema_inconc`	
8		`? TIMEOUT T_error_com`	
10		`+ Cerrar_sistema_inconc`	

Figura 11.1: Paso de Prueba que procesa posibles eventos en la espera de la confirmación de la primitiva `GET_PARAMETER_REQ`.

Se ha definido un Paso de Prueba para cada primitiva de confirmación, el cual es invocado inmediatamente después de la sentencia de envío de la primitiva. Todos estos Pasos de Prueba tienen la misma estructura y procesan tanto errores en la comunicación con el Subsistema Inferior, veredicto de inconcluso, como la recepción de valores inesperados (Figura 11.1).

Default Name		`Check_T_global (T_global:TIMER)`	
Group			
Objective			
Default		`Chequea el temporizador global`	
Comments			
Nr	**Label**	**Behaviour Description**	**Constraints Ref**
1		`? TIMEOUT T_global`	
2		`? OTHERWISE`	

Figura 11.2: Paso de Prueba por Defecto.

Además, hay un Paso de Prueba por Defecto común para todos los Casos de Prueba (Figura 11.2) que recoge los eventos inesperados que pueden darse a lo largo de la

SPICUM servicio de publicaciones

ejecución de la Prueba y procesa el vencimiento del temporizador global de la Prueba; este temporizador se pasa como parámetro y permite limitar la duración de la Prueba. En ambos casos se asigna un veredicto parcial de inconcluso.

La comunicación con el Subsistema Inferior se realiza a través de un único PCO. Tanto la Unidad de Señalización como el EUT son controlados manualmente por el operario, el cual también se encarga de conectar adecuadamente todos los equipos.

11.2 Sistema de Pruebas Radio para Bluetooth

Las pruebas radio para Bluetooth [BTEST RF] determinan la conformidad del EUT en aspectos como la potencia transmitida y su densidad espectral, la modulación empleada (2-GFSK[2]), las emisiones fuera de banda y la recepción. Los procedimientos de medida se describen en [EN 300 328] y algunas Pruebas pueden durar hasta dos horas como, por ejemplo, los Casos de Prueba `TRM/CA/06` o `RCV/CA/02`. Se han implementado 13 pruebas radio (Tabla 11.1) ([GOME01b], [LOBA02]). El conjunto de funciones TSO utilizado en este Juego de Pruebas se lista en la Tabla 11.2.

Tabla 11.1: Lista de Casos de Prueba radio de Bluetooth implementadas.

Identificador	Nombre	Descripción
`TRM/CA/01/E`	Potencia de salida	Verificación de salida de máxima potencia y potencia media.
`TRM/CA/02/E`	Densidad de potencia	Verificación de salida de máxima densidad de potencia.
`TRM/CA/03/E`	Control de potencia	Verificación del control de potencia de transmisión.
`TRM/CA/04/E`	Espectro de salida – Rango de frecuencia	Verificación de que las emisiones dentro del rango de frecuencias de operación están dentro de los límites.
`TRM/CA/05/E`	Espectro de salida – Ancho de banda a 20 dB	Verificación de que las emisiones dentro del rango de frecuencias están dentro de los límites.
`TRM/CA/06/E`	Espectro de salida – Potencia del canal adyacente	Verificación de que las emisiones dentro del rango de frecuencias de operación están dentro de los límites.
`TRM/CA/07/E`	Características de modulación	Verificación del índice de modulación.
`TRM/CA/08/E`	Tolerancia de frecuencia a la portadora inicial	Verificación de la precisión de la transmisión de frecuencia de portadora.
`TRM/CA/09/E`	Desvío de frecuencia de portadora	Verificación de la deriva de la frecuencia de portadora a lo largo del intervalo correspondiente a un paquete.
`TRC/CA/01/E`	Emisión de espurios fuera de banda	Verificación de que los espurios emitidos fuera de banda están dentro de los límites.
`RCV/CA/01/E`	Sensibilidad – Paquetes de una ranura	La sensibilidad es probada usando un transmisor (con paquetes de una sola ranura). El EUT debe conocer la sensibilidad requerida para la señal transmitida.
`RCV/CA/02/E`	Sensibilidad – Paquetes multiranura	La sensibilidad es probada usando un transmisor (con paquetes de múltiples ranuras). El EUT debe conocer la sensibilidad requerida la señal.
`RCV/CA/06/E`	Nivel máximo de entrada	Verificación de las prestaciones del receptor.

[2] *Gaussian Frequency Shift Keying*.

SPICUM servicio de publicaciones

Tabla 11.2: Lista de funciones TSO del Juego de Pruebas radio de Bluetooth.

TSO	Descripción
`TSO_Access_Code`	Obtiene la secuencia binaria de la palabra de sincronismo del código de acceso de un paquete Bluetooth.
`TSO_BER`	Devuelve en formato `IA5String`, el valor de la BER.
`TSO_Calc_Delta_Frec`	Determina el valor de las características de modulación.
`TSO_Calc_Deltafrec_Within`	Obtiene el ancho de banda 20 dB de la señal.
`TSO_Calc_Frec_Within`	Obtiene los valores de las frecuencias para los que la densidad espectral de la señal cae por debajo de –80 dB/Hz.
`TSO_Calc_Potencia_Media`	Calcula la potencia media de salida, devolviendo como resultado un elemento `IA5String`.
`TSO_Comparar_Valores`	Realiza la comparación entre dos valores reales, pasados como cadenas de caracteres (`IA5String`).
`TSO_Correlacion`	Obtiene la posición en la que se encuentra la secuencia dentro de la traza `y`.
`TSO_Division`	Divide dos números representados como cadenas `IA5String`.
`TSO_Estado`	Proporciona información acerca de la situación en la que se encuentra el Sistema de Pruebas durante la realización de la medida de la BER.
`TSO_Filtrado_Gaussiano`	Filtro que da forma de pulso gaussiano a una secuencia Bluetooth.
`TSO_Frecdrift`	Calcula la desviación en frecuencia.
`TSO_Icft`	Integra los cuatro primeros bits del preámbulo del paquete Bluetooth.
`TSO_Maxdriftrate`	Calcula la tasa de desviación máxima en frecuencia.
`TSO_Maxfrecdrift`	Calcula la máxima desviación en frecuencia.
`TSO_Producto`	Multiplica dos números representados como cadenas `IA5String`.
`TSO_Suma`	Suma o resta de números reales representados como cadenas `IA5String`.
`TSO_Traza_Ajustada`	Establece cuándo una traza ha sido ajustada correctamente para la búsqueda de espurios.

La instrumentación empleada para las pruebas de transmisión ha sido un analizador de espectros, modelo FSIQ26 [FSIQ26], y para las pruebas de recepción un generador de señal, modelo SMIQ03B [SMIQ03B], y un modulador I/Q, modelo AMIQ [AMIQ]. El EUT debe activarse en modo de transmisión salvo en las pruebas de recepción, en las cuales debe activarse en modo bucle (*loopback*).

El nivel de referencia para medidas de potencia debería ser 1 dB por encima de la potencia máxima que el EUT puede proporcionar (Tabla 11.3). Sin embargo, para evitar errores significativos de medida, se recalcula el nivel de referencia buscando el valor de pico de la señal, al que se suma 0,5 dB para fijar el nivel de referencia definitivo.

Tabla 11.3: Nivel de referencia a utilizar según la categoría del EUT.

Clase	Nivel de referencia
1	21 dBm
2	5 dBm
3	1 dBm

SPICUM servicio de publicaciones

11.3 Sistema de Pruebas Radio para UMTS

Las pruebas radio de UMTS [3GPP 34.121] se encargan de verificar la conformidad del EUT en sus aspectos de transmisión, recepción, rendimiento y gestión de recursos radio. Los procedimientos y configuraciones de medida se describen en [3GPP 34.108]. Se han implementado 14 pruebas de transmisión (Tabla 11.4) del modo FDD [VALE02]. El conjunto de funciones TSO utilizado en este Juego de Pruebas se lista en la Tabla 11.5.

Tabla 11.4: Lista de Casos de Prueba radio de UMTS implementadas.

Identificador	Nombre	Descripción
TRM/02	Nivel de Potencia – Máxima Potencia de Salida	Verificar el error de la potencia máxima de salida del UE[3].
TRM/03	Frecuencia – Error de Frecuencia	Verificar que el error de la frecuencia portadora del UE no excede de ±0,1 ppm.
TRM/04	Nivel de Potencia –Potencia Dinámica de Salida en el UL	-Grupo-
TRM/04/01	Nivel de Potencia – Control de Potencia en Lazo Abierto en el UL	Verificar la tolerancia del control de potencia en lazo abierto del UE.
TRM/04/02	Nivel de Potencia – Control de Potencia en Lazo Interior en el UL	Verificar que el UE ajusta su potencia de salida a los comandos recibidos en el enlace descendente.
TRM/04/03	Nivel de Potencia – Mínima Potencia de Salida	Verificar que la potencia de transmisión mínima del UE es inferior a –50 dBm.
TRM/04/04	Nivel de Potencia – Gestión de Potencia de Salida en Estado no Sincronizado	Verificar que el UE monitoriza la calidad DPCCH (*Dedicated Physical Control CHannel*) y enciende o apaga su transmisor adecuadamente.
TRM/05	Transmisión de Potencia en Estados ON/OFF	-Grupo-
TRM/05/01	Nivel de Potencia – Transmisión de Potencia en Estado OFF	Verificar que la potencia de transmisión OFF del UE es inferior a –56 dBm.
TRM/05/02	Nivel de Potencia – Transmisión en Estado ON/OFF en Máscara Temporal	Verificar los niveles de potencia de transmisión ON/OFF frente al tiempo.
TRM/06	Nivel de Potencia – Cambio de TFC	Verificar que el UE modifica su potencia de salida al cambiar la velocidad de transmisión.
TRM/08	Espectro – Ancho de Banda Ocupado (OBW – *Occupied BandWidth*)	Verificar que el ancho de banda ocupado de canal es menor de 5 MHz basado en una tasa de chip de 3,84 Mcps.
TRM/09	Espectro – Máscara de Emisión Espectral	Verificar la potencia de emisión del UE.
TRM/10	Espectro – Relación de Potencia Filtrada en el Canal Adyacente (ACLR - *Adjacent Channel Leakage Power Ratio*)	Verificar el ACLR del UE debido a modulación.
TRM/11	Espectro – Emisiones Espurias	Verificar las emisiones de espurios del UE.
TRM/12	Espectro – Transmisión de Intermodulaciones	Verificar la intermodulación en transmisión.
TRM/13	Espectro – Modulación de Transmisión	-Grupo-
TRM/13/01	EVM – Magnitud del Vector de Error (EVM - *Error Vector Magnitude*)	Verificar que el EVM no excede el 17,5% en la configuración indicada.

SPICUM servicio de publicaciones

La instrumentación empleada ha sido un analizador de espectros, modelo FSIQ26 [FSIQ26], complementado con un generador de onda continua, modelo ESG-D2000 [ESG-D2000], para generar ondas interferentes. Las Pruebas se realizan sobre un canal de referencia ascendente de tipo DPCH (*Dedicated Physical CHannel*) con una velocidad de usuario de 12,2 kbps y una tasa de bit ascendente a nivel físico de 60 kbps. Debe establecerse la llamada según el procedimiento genérico y la configuración indicada en la Sección E.3.1 de [3GPP 34.121]. El EUT debe activarse en modo bucle (*loopback*). Al igual que en Bluetooth, la máxima potencia de salida depende de la clase del EUT y se ajusta de forma similar.

Tabla 11.5: Lista de funciones TSO del Juego de Pruebas radio de UMTS.

TSO	Descripción
`TSO_Calc_ Trazax`	Mediante la cadena de caracteres que se obtiene del subsistema inferior se calcula el número de valores devueltos y se generan y devuelven los valores del eje `x` (de tipo `IA5String`).
`TSO_Calc_ACLR`	Comprueba que los canales adyacentes están dentro de los límites.
`TSO_Calc_Gap_Pretrig`	Se calcula el margen previo del disparo (*gap sweep pretrigger*) para obtener en pantalla el primer preámbulo.
`TSO_Calc_Mask_Preamb`	Determina si el preámbulo se encuentra dentro de la máscara.
`TSO_Calc_OBW`	Calcula el ancho de banda ocupada.
`TSO_Calc_OutPow`	Calcula la potencia promedio de salida.
`TSO_Calc_OutSyncHand`	Comprueba que se sigue un determinado patrón de potencia emitida por el móvil, y que está dentro de la máscara.
`TSO_Calc_Paso`	Comprueba que la traza capturada cumple con los requisitos que se piden (nivel de potencia).
`TSO_Calc_Pot_Preamb`	Calcula la potencia promedio del preámbulo.
`TSO_Calc_Ppm`	Obtiene el error de frecuencia en partes por millón (ppm).
`TSO_Calc_SpecEmissMask`	Determina si cumple la máscara de emisión espectral.
`TSO_Calc_Triggerlevel`	Calcula el nivel del disparo que debe de tener el analizador para obtener un resultado correcto en el siguiente paso de la prueba.
`TSO_Calc_Trm_Intermod`	Comprueba que se cumple tanto el nivel de la señal interferente, como el del producto de intermodulación.
`TSO_Comparar_Valores`	Realiza la comparación entre dos valores reales, pasados como cadenas de caracteres (`IA5String`).
`TSO_Compr_Espureos`	Comprueba que los valores de potencia no sobrepasan el nivel máximo permitido.
`TSO_Producto`	Multiplica dos números representados como cadenas `IA5String`.
`TSO_Obtener_Valor`	Cálculo del error de frecuencia o de EVM.
`TSO_Suma`	Suma o resta dos números representados como cadenas `IA5String`.

11.3.1 Ejemplo Detallado

Se describe la implementación del Caso de Prueba `TRM/09` de UMTS, que verifica que la emisión del EUT cumple los límites de la máscara de emisión espectral establecida. Para ello, la potencia de salida se mide a distintas frecuencias y se compara con la máscara de potencia de referencia. Este Caso de Prueba sólo necesita un analizador de espectros. El procedimiento de prueba realiza los siguientes pasos:

1. Enviar continuamente comandos de control de potencia de subida al EUT hasta que su potencia de salida alcanza su máximo nivel.

SPICUM servicio de publicaciones

2. Medir la potencia de la señal transmitida con el filtro de medida especificado. La frecuencia central del filtro se desplaza y se mide la potencia en cada punto. El ancho de banda del filtro es 30 kHz o 50 MHz dependiendo de la distancia a la frecuencia central.
3. Calcula la relación entre la potencia medida y la máscara de referencia.

Test Case Name		TC_TRM08_SpecEmissMask
Group		TRM/08/
Purpose		Verificar que la mascara espectral de emisión se cumple para distintas variaciones de la frecuencia portadora, tanto para altas como bajas frecuencias
Configuration		
Default		Check_T_global_trm08
Comments		
Selection Ref		TCS_TRM08
Description		Verificación de la mascara de emisión espectral para bajas y altas frecuencias.
Nr	**Label**	**Behaviour Description**
1		START T_global_trm08
2		+Init_system
3		+Inic_an_esp_trm08_ftx_low
4		+Calc_SpecEmissMask_low
5		+EUT_ftx_high
6		+Inic_an_esp_trm08_ftx_high
7		+Calc_SpecEmissMask_high
8		+Check_res_trm08

Figura 11.3: Código principal para el Caso de Prueba `TRM/09`.

La Figura 11.3 muestra el nivel superior del Caso de Prueba. Las acciones que se realizan son las siguientes:

1. Se inicializa la instrumentación (`INIT_INSTRUMENT`) y se crea la configuración de medida (`CONNECT`).
2. Se configura el filtro de medida en el analizador de espectros (`SET_PARAMETER`) con un ancho de banda de 30 kHz.
3. Se mide la potencia de pico (`GET_PARAMETER`).

Test Step Name		Calc_SpecEmissMask_low	
Group		Calc_Steps/TC_TRM09/	
Objective			
Default		Check_T_global(T_trm09)	
Comments			
Nr	**Label**	**Behaviour Description**	**Constraints Ref**
21		(TCV_freq_stop := TSO_RESTAR(TSC_fx_low, TSC_par_span_2500k))	
22		GPIB!SET_PARAMETER_REQ	Set_parameter_req(TSC_id_an, TSC_com_stop, TCV_freq_stop)
23		+Set_parameter_err_rsp	
24		START T_wait_1_s	
25		?TIMEOUT T_wait_1_s	
26		GPIB!GET_PARAMETER_REQ	Get_parameter_req(TSC_id_an, TSC_com_get_trace, TSC_par_trace)
27		+Get_parameter_err_rsp	
28		(TCV_banda_izq := TCV_par)	

Figura 11.4: Código para obtener la medida de potencia en la banda.

SPICUM servicio de publicaciones

4. Se toman medidas (`GET_PARAMETER`) de la potencia en la banda ±2,5 MHz alrededor de la frecuencia portadora, f_c, a intervalos de 5 kHz. Primero se miden las frecuencias en la banda superior; el código para obtener esta medida se muestra en la Figura 11.4.
5. Se configura nuevamente el analizador de espectros y se toman medidas para la banda inferior.
6. Se configura el analizador de espectros (`SET_PARAMETER`) para una resolución de ancho de banda de 50 kHz.
7. Se toman medidas (`GET_PARAMETER`) de la potencia en la banda ±12,5 MHz alrededor de la frecuencia portadora, f_c, a intervalos de 5 kHz, salvo en la banda empleada anteriormente.
8. Las medidas obtenidas se comparan con la máscara de emisión de referencia y se decide el veredicto.

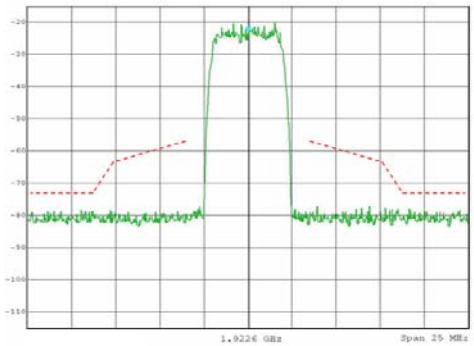


Figura 11.5: Resultado gráfico del Caso de Prueba de la Máscara de Emisión Espectral.

La Figura 11.5 muestra un ejemplo de las medidas obtenidas en esta Prueba. La línea punteada representa los niveles de potencia de referencia. La Figura 11.6 muestra la secuencia de mensajes intercambiados entre el Subsistema de Pruebas y el Subsistema Inferior desde el comienzo de la ejecución hasta el cuarto paso, donde se obtiene la primera medida.

11.4 Conclusiones

En este capítulo se ha demostrado la aplicación de la Metodología de Diseño al ámbito de los Sistemas de Pruebas radio. El uso de una misma notación que para las pruebas de protocolos permite reutilizar los componentes de la arquitectura y reducir el coste de diseño y mantenimiento. Para el control de la Instrumentación se ha utilizado el bus GPIB, junto con el estándar de comandos SCPI, por su universalidad.

Como ejemplos de aplicación se han diseñado Sistemas de Pruebas radio de Bluetooth y UMTS. Se ha descrito en detalle el modelado de un Caso de Prueba de UMTS. Con ello, damos por terminada la descripción de los Sistemas de Pruebas que se han construido siguiendo la Arquitectura y la Metodología de Diseño presentadas en los primeros capítulos.

SPICUM servicio de publicaciones

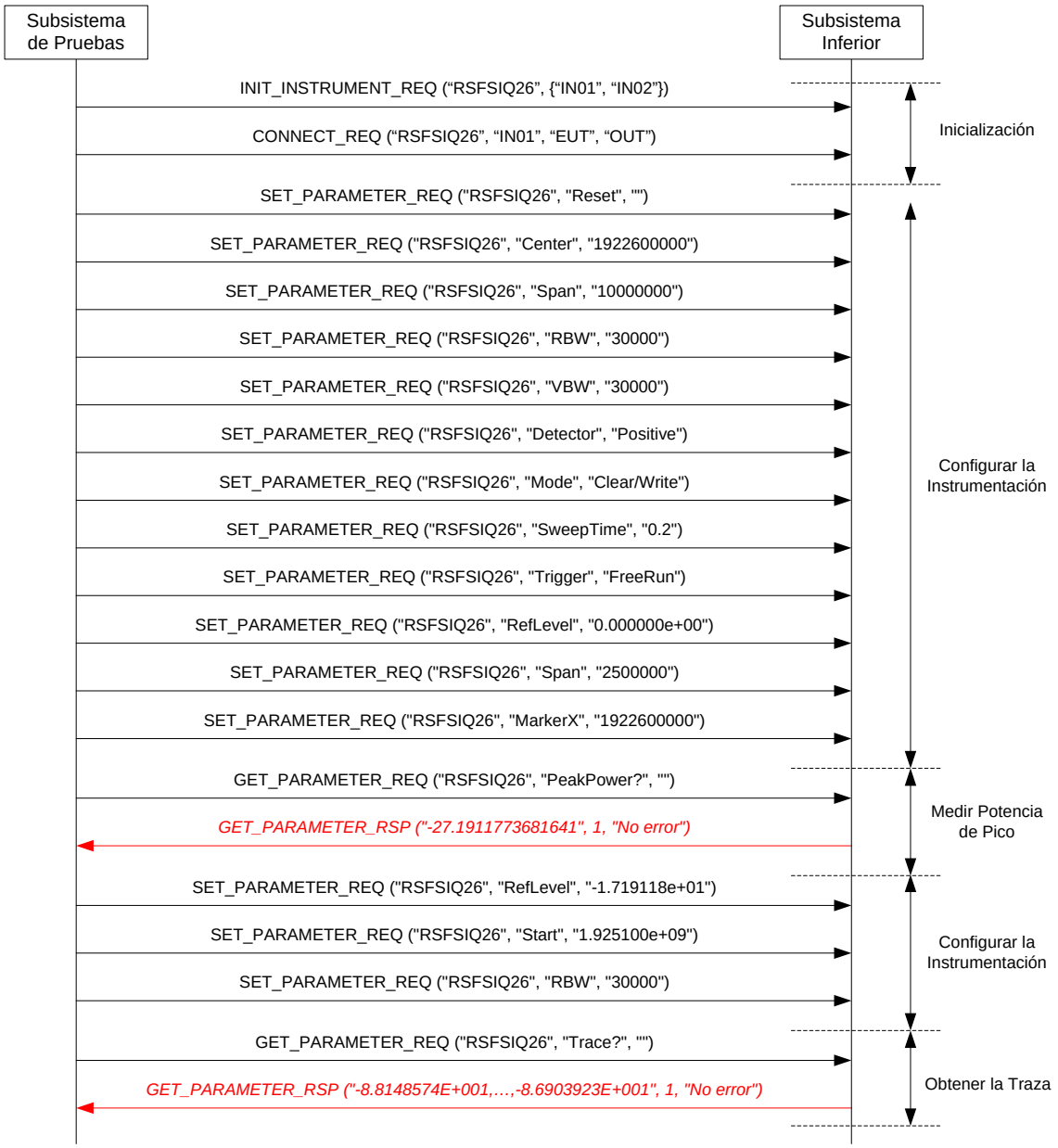


Figura 11.6: Secuencia de mensajes para el Caso de Prueba Máscara de Emisión Espectral hasta que se realiza la primera medida[4].

SPICUM servicio de publicaciones

SPICUM
servicio de publicaciones

CONCLUSIONES

SPICUM
servicio de publicaciones

SPICUM
servicio de publicaciones

Capítulo

12

"¡Saboreemos nuestra locura!
¡El hombre nació para ser feliz!

Sus vanas pretensiones,
y vanas defensas,
turban sus sentidos, obnubilan su mente.

Delgados o gordos,
retozamos y adulamos,
así que regocijémonos y finjamos.

¡La diversión es el triunfo
de la mente sobre la materia,
y todos llegaremos a casa si reímos al final!"

Emerson d'Anite, Los Límites del Cielo

CAPÍTULO 12: CONCLUSIONES

En esta Tesis se ha desarrollado una Metodología de Diseño de Sistemas de Pruebas válida tanto para pruebas de protocolo como para pruebas de capa física. El proceso de diseño elaborado se ha basado en un conjunto de principios básicos: uso de lenguajes y notaciones ITU, independencia de la plataforma de ejecución y uso de herramientas comerciales.

Los Sistemas de Pruebas son sistemas complejos, donde un diseño eficiente requiere una adecuada planificación y estructuración de las actividades. Para reducir la complejidad, se ha definido una arquitectura general, identificando los componentes necesarios y asignándoles responsabilidades, y un conjunto de herramientas, que simplifican el desarrollo formando parte del diseño o facilitando la generación de algún componente de la arquitectura. Las aportaciones más importantes se enumeran a continuación.

La Metodología de Diseño estructura el proceso de desarrollo en ocho fases, desde la definición hasta la obtención de un Sistema de Pruebas ejecutable en las plataformas deseadas. Cada fase engloba una o más actividades para cada una de las cuales se han indicado los modelos y documentos de entrada a partir de los cuales se generan los modelos y documentos de salida, que son utilizados en fases posteriores. Las dependencias entre actividades se han tratado de mantener dentro de una progresión

SPICUM servicio de publicaciones

lineal, aunque el proceso de diseño es básicamente iterativo. Tomando como base esta metodología, es posible realizar una planificación más precisa del proceso de desarrollo.

La Metodología asume un diseño basado en lenguajes de la familia ITU, como son SDL, TTCN y ASN.1. Estos lenguajes permiten realizar un diseño abstracto de alto nivel. La adaptación a una plataforma específica de ejecución se realiza a través de componentes externos al modelo del comportamiento de las pruebas o de los protocolos. La Metodología no requiere una herramienta comercial específica, y así se ha demostrado con el uso de entornos de desarrollo distintos. No obstante, su utilización práctica exige tener en cuenta las características de los entornos de desarrollo disponibles, lo que ha influido en algunas de las decisiones de diseño.

La fase menos concretada corresponde al diseño de los Módulos de Protocolo integrados en el Sistema de Pruebas, donde sólo se han indicado algunas posibles pautas de diseño a muy alto nivel. La realización de estos módulos es un proceso de diseño software, para lo cual existen ya diversas metodologías de desarrollo. En este trabajo se ha evaluado la metodología SOMT, que sugiere un conjunto de actividades para el modelado de sistemas software mediante el uso de SDL. Se ha estudiado cómo se ven afectadas las actividades que propone la metodología SOMT cuando se diseñan sistemas basados en estándares. Así, parte de las actividades propuestas dejan de ser necesarias, por haberse realizado en la elaboración del propio estándar. Como resultado, se ha formulado una nueva metodología, M-SOMT, que simplifica el conjunto de actividades a realizar y modifica ligeramente el objetivo de algunas de las restantes. M-SOMT ha sido aplicada al modelado del Nivel de Red del Sistema DECT.

La arquitectura definida está compuesta por componentes genéricos, reutilizables sin modificación en diferentes Sistemas de Pruebas, y componentes específicos, que deben ser implementados en cada ocasión. Esto permite disponer de un mismo marco de desarrollo y de operación para todo tipo de pruebas. El control y la gestión de las Pruebas se han independizado del comportamiento de las mismas y, a su vez, los módulos que realizan este comportamiento se han separado de los protocolos de comunicación subyacentes. Esta modularidad permite múltiples configuraciones en el despliegue y la selección de la plataforma de ejecución más adecuada para cada componente.

La arquitectura es independiente de la plataforma de ejecución e internamente se ha estructurado para que toda la funcionalidad dependiente de estas plataformas se encuentre agrupada en el mismo módulo. Adaptando este módulo a la nueva plataforma, es posible portar el Sistema de Pruebas sin modificar el resto de elementos. Como ejemplo, se han utilizado las plataformas Linux, Windows, Unix y DSP/BIOS.

Un diseño eficiente requiere la utilización de herramientas que permitan realizar un diseño a alto nivel y que automaticen parte de las tareas. Allí donde las herramientas comerciales no ofrecen un soporte adecuado, se han desarrollado herramientas propias. Estas herramientas se pueden clasificar en las categorías de componentes reutilizables, que forman parte de la arquitectura, y generadores automáticos, que permiten generar interfaces entre componentes de la arquitectura. Las herramientas propias están adaptadas a las características de las herramientas comerciales, ofreciendo así un entorno de desarrollo conjunto.

Para validar la Metodología de Diseño, se han construido Sistemas de Pruebas de conformidad de protocolos para los sistemas DECT, Bluetooth y UMTS. En los dos últimos casos el resultado han sido sendos Sistemas de Pruebas comerciales. En el caso de Bluetooth, algunos de los desarrollos se han realizado con un entorno comercial

SPICUM servicio de publicaciones

distinto al empleado en las demás implementaciones. Los resultados aplicables a las pruebas de conformidad son también aplicables a las pruebas de interoperatividad. Esto se ha demostrado con el desarrollo de Sistemas comerciales de Pruebas de interoperatividad para protocolos de Bluetooth y UMTS. Estos Sistemas de Pruebas han sido portados a distintas plataformas.

Aunque la Metodología de Pruebas de conformidad no cubre el área de las pruebas de conformidad de capa física la especificación de las pruebas en ambos campos se realiza bajo las mismas pautas. Por ello, se ha aplicado la Metodología de Diseño al desarrollo de Sistemas de Pruebas de capa física, utilizando la misma arquitectura que para los Sistemas de Pruebas de protocolos. De esta forma, se ofrece un mismo entorno de operación independientemente del tipo de prueba que se ejecute. Se ha demostrado que es posible modelar las pruebas de capa física mediante la notación TTCN, el estándar para la especificación de las pruebas de protocolo, y que este modelado puede ser independiente de la instrumentación de medida que se elija. Para ello, se ha definido una interfaz genérica entre las pruebas de capa física y la instrumentación. Este podría ser un camino para que las pruebas de capa física igualen el grado de formalización que ya ofrecen las pruebas de protocolos. Se han construido Sistemas de Pruebas de conformidad de capa física para Bluetooth y UMTS.

12.1 Líneas Futuras

El trabajo presentado en esta tesis puede continuarse por varios caminos, buscando mejorar la eficiencia del proceso de diseño e incorporando las presumibles futuras características que los Sistema de Pruebas deberán ofrecer.

Sobre lenguajes de modelado. La Metodología de diseño se puede adaptar a los últimos lenguajes y notaciones que han surgido tanto en el área de pruebas como en el diseño de sistemas en general. Me estoy refiriendo más concretamente a las notaciones TTCN-3 y UML. En general, para poder utilizar TTCN-3 en el proceso de diseño, las interfaces de cada uno de los componentes de la arquitectura no sufren modificaciones semánticas, aunque sí sintácticas, pero es necesario adaptar las herramientas a las características del código generado por los entornos de desarrollo.

Por otra parte, sería interesante adaptar los modelos y documentos empleados a lo largo de la Metodología de Diseño a la notación UML, que se encuentra ya soportada por entornos de desarrollo con suficiente funcionalidad. En este punto, no cabe duda de que una mayor integración entre SDL y UML, como está intentando ITU, será muy positiva, empleando UML para las tareas de definición y documentación y usando SDL para la descripción del comportamiento. Todavía faltan entornos de desarrollo que posibiliten un proceso de diseño basado en ambas notaciones de una forma realmente integrada.

Sobre plataformas de ejecución. En este trabajo se ha asumido que la plataforma es suficientemente potente, permitiendo realizar el diseño sin que las exigencias de prestaciones del mismo sean excesivas. Sin embargo, pensando en la portabilidad de estos sistemas, en el sentido de movilidad geográfica, este aspecto pasa a ser verdaderamente relevante. En esta situación el consumo y el peso son factores esenciales. Por ello, una mejora del proceso de diseño sería estudiar posibles reducciones en el tamaño del código generado por los entornos de desarrollo, para disminuir la memoria necesaria, y su optimización en velocidad, para poder utilizar procesadores de menor consumo. Un pequeño prototipo se ha realizado portando a una plataforma DSP los Sistemas de Pruebas para DECT.

SPICUM servicio de publicaciones

Sobre pruebas de capa física. En el campo de las pruebas de capa física el énfasis debería ponerse en alcanzar un grado de formalización de las Especificaciones de Prueba similar al ofrecido para las pruebas de protocolo. Esto exige buscar paradigmas de modelado que puedan ser aceptados tanto por suministradores como por laboratorios. Para ello, será necesario definir una interfaz adecuada de las pruebas, para lo que se ha ofrecido una propuesta en este trabajo, y acordar mecanismos para la configuración dinámica de la instrumentación de medida.

Sobre pruebas distribuidas. Las pruebas de sistemas están encaminándose hacia un enfoque distribuido, donde un componente central asigne el veredicto final pero la interacción con la Implementación Bajo Prueba se realice a través de múltiples componentes de prueba ubicados en diferentes nodos de la red. Modificar la arquitectura utilizada en este trabajo es relativamente simple, ya que la interacción entre los componentes de prueba es gestionada por el código que genera el entorno de desarrollo. Para la codificación de la información se pueden utilizar las mismas funciones que para la comunicación con el entorno de las pruebas. El Módulo de Protocolos pasa a estar también distribuido, y sería necesario crear el número de instancias requerido, que pueden ser del mismo o de distinto tipo.

SPICUM servicio de publicaciones

SUMMARY
(IN ENGLISH)

SPICUM
servicio de publicaciones

SPICUM
servicio de publicaciones

METHODS AND TOOLS FOR THE DESIGN AND VERIFICATION OF MOBILE COMMUNICATIONS SYSTEMS

1 Overview

Abstract

The aim of this thesis has been the procurement of technology for efficient design and maintenance, both in terms of time and economic cost, of Test Systems for wireless communications equipment. The proposed design process has been based on a set of basic principles: use of ITU languages and notations, independence from the execution platform and use of commercial tools.

This thesis describes:

- A generic architecture for Test Systems.
- A Design Methodology for Test Systems which identifies the stages to be followed and the documents and models which each stage should generate.
- A set of support tools for the methodology which, together with commercial tools, provide a development environment that covers the needs of all stages.
- The application the architecture, methods and tools used in protocols to Test Systems for the physical layer.
- A proposed methodology for SDL modeling of systems based in standards.
- A set of implementations that validate the proposed methodology and the tools that have been implemented.

In the initial conception of this work, the approach taken was oriented towards Test Systems for the certification of protocols within the framework of the OSI Methodology for Conformance Testing. As work proceeded the methodology has been found equally valid for other approaches. The results applicable to conformance testing are also applicable to the interoperability testing of protocols. Furthermore, the results that have been obtained for the area of testing protocols have been also applied to the field of radio tests cases, showing that it is possible to use the same architecture, method and tools in both cases, thereby reducing development costs.

To validate the design methodology, conformance protocol Test Systems for DECT, Bluetooth and UMTS have been built on different platforms. In the latter two cases the resulting Test Systems have been commercialized. The application of the methodology in the areas of radio testing and interoperability testing has been demonstrated for Bluetooth and UMTS systems.

Generic Architecture

A Test System can be thought of as a set of basic functions, common to any Test System, plus a group of more specific functions. The defined architecture is composed of generic components, reusable without modification in different Test Systems, and

SPICUM servicio de publicaciones

specific components, that must be implemented on each occasion. This allows to using the same development and operation framework for all types of tests. The control and management of the test cases have been separated from their behavior and, at the same time, the test cases have been separated from the underlying communication protocols (called Lower Subsystem). Also, the abstract behavior of the test cases and of the Lower Subsystem protocols has been separated from those functions that allow their execution on a specific platform; this module has been called Platform Adaptation Module and contains required low-level functionality such as I/O, time and memory management. The modularity of the architecture allows for multiple deployment configurations and selecting the most appropriate execution platform for each component.

Design Methodology

The Design Methodology structures the development process in eight stages from the definition of the Test System until the production of a Test System which can be executed on the desired platforms. Each phase includes one or more activities; for each of them it has been specified the input models and documents from which the output models and documents, to be used in later stages, are generated. It has been tried to keep dependencies between activities within a linear progression, but the design process is essentially iterative. Based on this methodology, it is possible to make a more precise planning of the development process.

The methodology assumes a design process based on languages of the ITU family such as SDL, ASN.1 and TTCN. SDL has been used for the design of the protocols used in the Test Systems, while it has been assumed that the test cases are modeled in TTCN. The interface between these elements is described in ASN.1. These languages allow a high level of design abstraction. The methodology does not require a specific commercial tool, and it has been demonstrated using different development environments. However, in practice, its use requires taking into account the characteristics of the development environments, which has influenced some of the design decisions.

The design of systems in SDL is just a case of the more general problem of system design software. This is a complex area in which there are several alternatives. Therefore, we evaluated a design methodology based on SDL, SOMT (SDL-oriented Object Modeling Technique) as a possible guide. When SOMT is applied to the design of standards-based systems, some of the activities suggested can be simplified. Thus, some of the proposed activities are no longer necessary, as they must have been performed along the development of the standard itself. As a result, it has been proposed a modified methodology (M-SOMT) for the design in SDL of standards-based systems that simplifies the set of activities to carry out and slightly modifies the target of some of the others. M SOMT has been applied to the modeling of the Network layer of DECT equipment.

Tools

Although the methodology is conceptually independent of specific tools and notations, and it has been demonstrated so, its use requires performing tasks that are only viable through the use of automated tools. These tools include editors, compilers, automatic generators and simulators. Developing and maintaining a proprietary tool represents a high cost for any company, so, to perform the various tasks, commercial tools have been used where possible. The shortcomings of the commercial tools have been filled with the design of our own tools, primarily in two areas: generic components of the architecture and automatic interface generators. Within the automatic generators, tools

SPICUM servicio de publicaciones

have been developed to encode and decode messages from the interface between the Test Case Subsystem and the Lower Subsystem components; they allow both the use of ASCII and PER transfer syntaxes. These tools have been tailored for the commercial development environments that have been used.

Implementations

The methodology has been applied to the design of Test Systems, both for conformance and interoperability areas, for protocol and radio testing. In the first instance, it was validated by building a prototype for DECT communications equipment; Test Systems have been built for the DLC and NWK layers of Portable and Fixed Terminations. The designed Test Systems have been ported to Linux, Solaris, Windows and DSP / BIOS.

Subsequently, it has been used for the design of commercial Test Systems for UMTS and Bluetooth equipment. The use of the Design Methodology in the Bluetooth technology has allowed, from the suggestions of the design group of the company, to improve the tools and components used. The resulting Test System, BITE (Bluetooth Qualification Tester), has been commercially distributed, and has turned out to be the best-selling worldwide protocol conformance Test System for Bluetooth.

The Bluetooth technology has also been the area where the flexibility of the methodology has been checked through the use of different development environments and the application of the proposed architecture to interoperability Test Systems. Systems have been built to test the LM and SPP layers in which the Test Case Subsystem has been generated using a different tool. The integration with the Lower Subsystem has required adapting the interfaces offered by the new tool; adjustments have been necessary on the codification functions of the interface with the Lower Subsystem. For some of these adaptations automatic tools have been built. On the other hand, in Bluetooth, conformance tests are only defined for the lower protocols, while for profiles, layers closer to the user, interoperability tests have been defined. Therefore, it has been evaluated the application of the design methodology to the design of an interoperability Test System, in particular for the Headset profile.

The design methodology has been applied to the implementation of a commercial UMTS protocol Test System (MINT). In particular, there has been a direct involvement in the design of the Lower Subsystem. Furthermore, progress has been made in formalizing the design of interoperability Test Systems, integrating commercial tools in the process and adapting our own tools. The result of the collaboration has also been used as signaling unit of a radio conformance Test System.

As communication systems have evolved, so has done the design methodology. Initially, in the DECT system, only the languages SDL and TTCN were used. Although an attempt was made to use ASN.1 to define the interface between the two models, it was not possible given the limitations of the tools. An ASN.1 to SDL converter was developed, but its maintenance was dropped when commercial development tools provided similar tools. The test cases for Bluetooth and UMTS equipment contain ASN.1 declarations, thus the methodology was changed in this regard. Similarly, the automatic interface generators were also adapted

Application to physical layer testing

As for protocols, all communication equipment operating in a regulated system must undergo a certification process for its physical interface. A characteristic difference between protocol and physical layer Test Systems is the fact that the latter require specific instrumentation able to perform certain actions on the electrical interface.

SPICUM servicio de publicaciones

Examples of this instrumentation are: signal generators, modulators, oscilloscopes, spectrum analyzers, etc. This instrumentation will typically have a fairly high cost, and since several equipment are usually required, this means that the physical layer Test Systems are considerably more expensive than protocol Test Systems.

The conformance testing methodology, as specifically indicated, does not fully cover the area of physical layer testing. Despite this, the certification of the physical interface of a communication system follows the same guidelines than for protocols layers. The difference is that protocol test cases are modeled in TTCN, while physical layer test cases are described textually.

Therefore, the methodology has been applied to the development of design systems for physical layer testing, using the same architecture as for protocol Test Systems. In this way, the same operation environment is provided, regardless of the type of test that is run. It has been shown that it is possible to model physical layer test cases using the TTCN, the standard for the specification of protocol test cases, and that this modeling can be independent of the chosen measurement instrumentation. In order to achieve this independence, a generic interface between the physical layer test cases and the instrumentation has been proposed. This could be a way for physical layer test cases to match the degree of formalization that is already provided by protocol test cases. Physical layer conformance Test Systems have been built for Bluetooth and UMTS.

The following sections describe in more detail the contents of this Thesis.

2 Conformance Testing

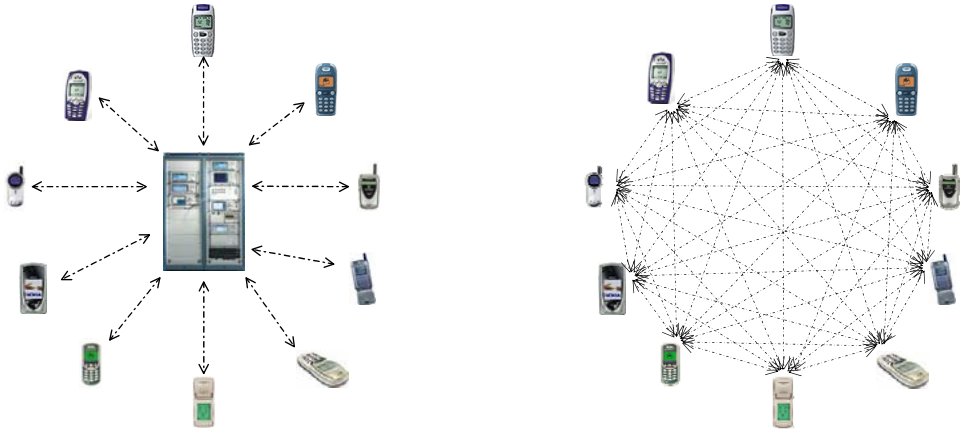
When it comes to verifying an implementation, there are two alternatives: conformance testing and interoperability testing. Conformance testing seeks, primarily, to certify the device behavior with respect to its specifications and determine that it does not interfere with the operation of other equipment, "providing a high probability of proper behavior at the interface between implementations from different suppliers" [EWOS96]. It is said that a system conforms to the specification if it meets the requirements, static and dynamic, of the applicable specifications in its communication with other systems [X.290].

Interoperability tests, on the other hand, check the ability of a computer to interact with others, relativizing the strict compliance with the specifications [LLOY89]. Interoperability is the “ability of two or more open systems to perform a common distributed task that is supported by OSI protocols OSI” [GTS 029].

1.2.1 Description of the Conformance Testing Methodology

The OSI conformance testing methodology [ISO 9646] standardizes the process of testing an implementation and includes both the test mechanisms as well as the documentation needed during the process. The solution adopted by ISO to specify the tests was set defining, for each protocol or set of them, a set of tests called Abstract Test Suite (ATS). Each test is called Abstract Test Case (ATC). For each Test Case in an Abstract Test Suite a result is obtained at the end of its execution. There are three possible verdicts: Pass, Fail and Inconclusive. To run an ATS, an Executable Test Suite (ETS) must be generated. The ETS is simply an executable version of the abstract model of the tests, properly adapted to the runtime platform.

SPICUM servicio de publicaciones



(a) (b)

Figure 1: Conceptual representation of a) conformance testing and b) interoperability testing.

An Abstract Test Method (ATM) specifies the test configuration, defines the interfaces through which the Test System can observe and influence the behavior of the IUT and provides a mechanism for coordinating the test execution.

When applying for an equipment certification, suppliers are required to declare the functionality of its implementation, known as the Implementation Under Test (IUT). From these statements, the test laboratories are responsible for selecting the appropriate set of test cases. As a result a report is generated granting or denying the conformance certification as stated by the verdicts obtained in the different test cases.

The OSI methodology defines a set of documents that can be grouped in:

- Test Specifications: Test Suite Structure & Test Purposes (TSS&TP), Abstract Test Suite (ATS), Profile Test Specification (PTS) y Profile Specific Test Specification (PSTS)
- Information of the Implementation Under Test: Implementation Conformance Statement (ICS), Implementation eXtra Information for Testing (IXIT) y System Conformance Statement (SCS).
- Test Reports: System Conformance Test Report (SCTR) y Protocol Conformance Test Report (PCTR).

3 Test Systems Architecture

The development of a Test System is a systems engineering problem that encompasses multiple disciplines: protocols, software, hardware, signal processing, radio. At a high level, a Test System can be viewed as a configurable runtime platform, which is adapted according to the characteristics of the test cases to execute. Much of the functionality can be seen as playing a support role, in the sense that it carries out tasks which are independent of the System Under Test. The proposed architecture for Test Systems is shown in Figure 2.

SPICUM servicio de publicaciones

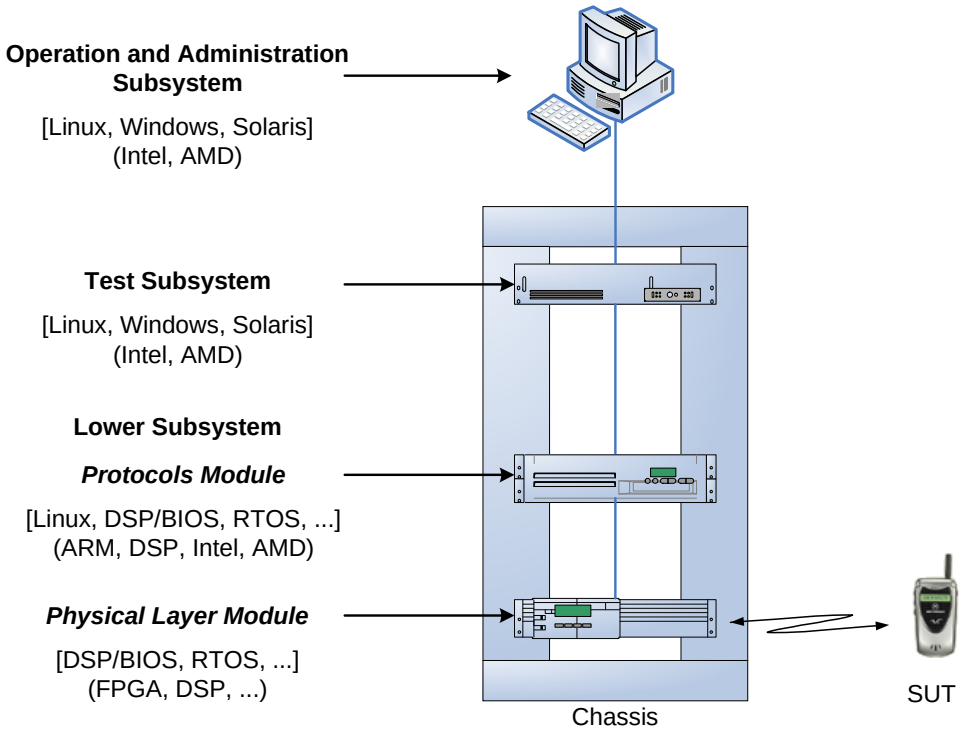


Figure 2: Flexible distribution of subsystems among physical platforms.

The structure of the Test Subsystem and the Protocols Module has been organized to be as similar as possible in both cases. Both can be modeled in an abstract language, i.e. independent of the platform. The Engine modules contain the libraries for the interpretation of the semantics. For the execution, it is necessary to adapt these abstract models to the chosen platform. The necessary elements have been grouped in an Adapter Module, which provides services typical of an operating system (memory, time, input/output and concurrency management) and a Management Module, which provides high level services (encryption, control, logging, ...).

The Operation and Administration Subsystem allows the operator to control the execution of the tests and manage their results. It consists of an Operation Interface, which is the graphical element through which the operator communicates with the Test System, and a Management Module, which contains the necessary functionality to implement, control and manage the tests and their results, as well as for generating test reports from them.

The Test Subsystem includes the tests and all the elements necessary for their execution. It represents the test executable. It consists of the Abstract Test Suite, the TTCN Engine, the Test Adapter Module and the Test Management Module. The Abstract Test Suite models the test behaviors, i.e. the sequence of stimuli and responses required to verify a specific test purpose. The end-to-end coder (Codec $E2E_{TTCN}$) handles the coding, and decoding, of PDUs exchanged between the tests cases and the IUT.

SPICUM servicio de publicaciones

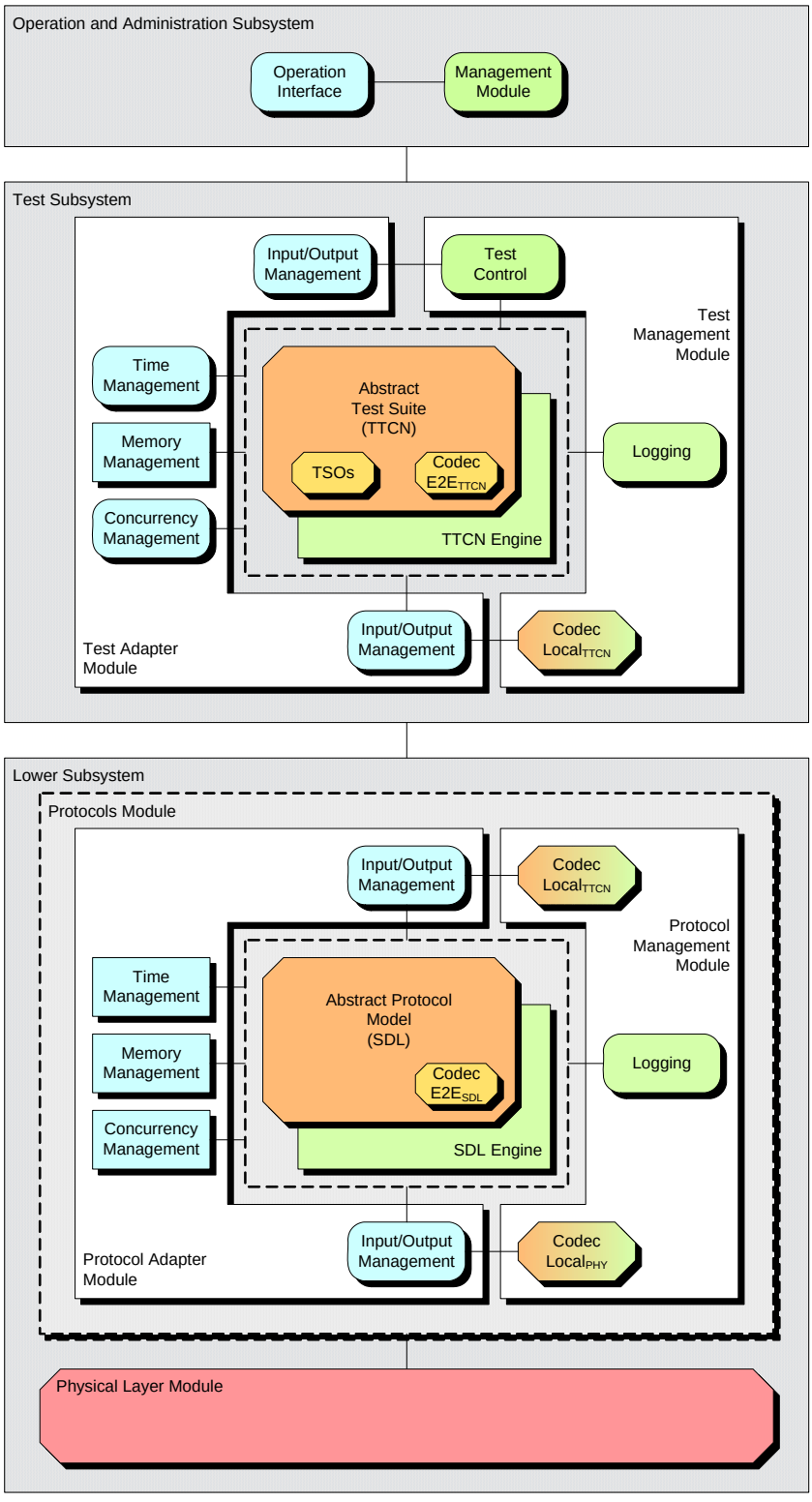


Figure 3: Detailed architecture of a Test System.

The Lower Subsystem contains the protocols and components needed to communicate the Test Subsystem with the System Under Test. Therefore, this subsystem has been separated in two modules, each covering a different design problem: Protocols Module and Physical Layer Module. The first one is more related to the telematic concept of protocol (sending and receiving messages, asynchrony, request-response), while the

SPICUM servicio de publicaciones

second refers more to signal processing (modulation, reception and transmission filters, time synchronization, frequency domain, ...). The separation of functionality between the two of them depends on the requirements of speed and time synchronization. The communication between the Test Subsystem the Physical Layer Module is always carried via the Protocols Module.

The Protocols Module consists of the Protocol Abstract Model, the SDL Engine, the Protocol Adapter Module and the Protocol Management Module. The Protocol Abstract Model represents the state machine modeling the abstract behavior of each of the protocols. The architecture does not restrict the language used for this modeling. As in the Test Subsystem, it requires end-to-end encoders (Codec $E2E_{SDL}$) for PDUs exchanged with the System Under Test.

The physical layer module is more akin to the area of hardware design than protocol design, so it has not been studied in this Thesis.

Figure 3 shows the architecture in detail. This architecture is valid for all kinds of tests, and it is independent of the runtime platform, as well as commercial tools. However, its practical use requires taking into account the capacities of the development environments, which has influenced some of the design decisions.

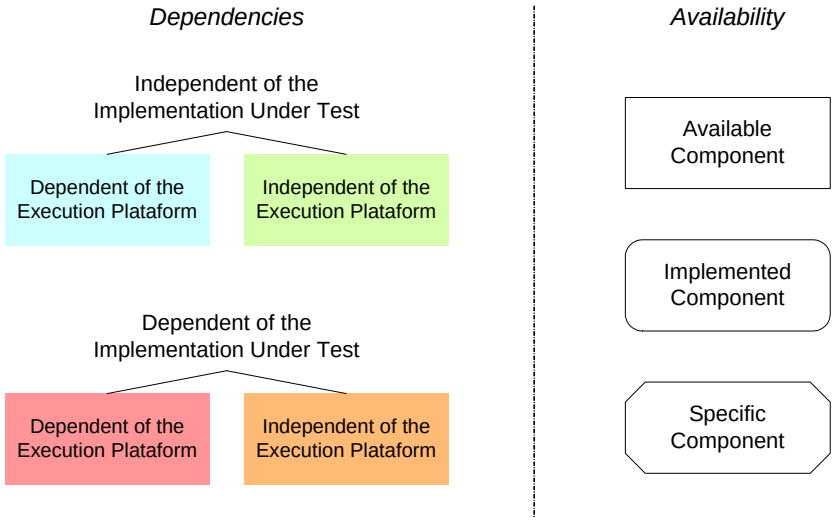


Figure 4: Legend of colors and shapes used to classify the components of a Test System.

To classify the components of a Test System a color and shapes code has been used. To make this classification three criteria have been taken into account: dependence on the Implementation Under Test, dependence on the runtime platform and availability at the beginning of the design. The first two criteria are shown in the figures of the architecture through colors (left side of Figure 4). The degree of availability is indicated by its shape (right side of Figure 4). We have distinguished between components previously available (standards, commercial tools), components that have been implemented (our own tools) and components that are specific to each Test System (abstract models, coders). The Management Module coders are shown with a color gradient because its implementation uses both parts independent of the IUT (generic) and parts dependent on the IUT (which are constructed for each Test System).

SPICUM servicio de publicaciones

Table 1: Availability of the components of the architecture of a Test System.

Subsystem	Module	Component	Available	Implemented	Specific
Operation and Administration Subsystem					
	Operation Interface			X	
	Management Module			X	
Test Subsystem					
	Abstract Test Suite				X
		TSOs			X
		Codec $E2E_{TTCN}$			X
	TTCN Engine		X		
	Test Adapter Module				
		Input/Output Management		X	
		Time Management		X	
		Memory Management	X		
		Concurrency Management		X	
	Test Management Module				
		Tests Control		X	
		Logging		X	
		Codec $Local_{TTCN}$		X	X
Lower Subsystem					
Protocols Module					
	Protocols Abstract Model				X
		Codec $E2E_{SDL}$			X
	SDL Engine		X		
	Protocol Adapter Module				
		Input/Output Management		X	
		Time Management	X		
		Memory Management	X		
		Concurrency Management	X		
	Protocol Management Module				
		Codec $Local_{TTCN}$		X	X
		Logging		X	
		Codec $Local_{PHY}$		X	X
Physical Layer Module					X

As part of this Thesis all the generic components that were not available have been built as well as tools for the automatic generation of coders between the Test Subsystem and the Lower Subsystem. Table 1 summarizes which of the components have been implemented, which were previously available and which are specific to each Test System. When designing a new Test System it is only necessary to build the latter ones.

SPICUM servicio de publicaciones

4 Design Methodology

The design is an engineering task which creates and develops a plan for a system, product, structure or component. The design process is composed of a multitude of activities, each of which can be viewed as a system that receives a set of models at its entry and produces a set of models at its output, result of applying on the inputs the processing defined in the activity. The moment when an activity is performed depends not only on when its inputs are ready, which, in general, are outputs of other activities, but also on the available resources, human and material.

This section describes a Design Methodology for Test Systems ([PONC99], [PONC00]) based on the use of formal languages and the architecture presented above. The methodology is independent of specific commercial tools.

1.4.1 Overview of the Design Methodology

The Design Methodology encompasses all activities necessary for the construction of a Test System from the system specifications and the test specifications. The activities of the methodology have been organized so that the design of the Test Subsystem and the Lower Subsystem can be done in parallel by different working groups. Figure 5 shows the phases that make up the methodology, indicating whether their results are text documents, models or executable entities. Although the activities are presented as a linear sequence, this structuring only aims to expose the dependencies between them. The design of these systems is an iterative process.

The methodology assumes, in a natural way, the use of languages and notations of the ITU family [AMYO99]. The Test Suites are modeled in TTCN ([X.292], [EN 201 873-1]), the Protocols Module are modeled in SDL ([Z.105], [Z.107]); the MSC notation [Z.120] notation is used as auxiliary.

This chapter does not specify a particular methodology for the design of SDL systems. What it is stated in this chapter is a sequence of activities that allow to structure the design of these systems and define milestones, outputs of each stage, in the process. The use of an SDL design methodology is discussed later.

The starting point of the Design Methodology is the set of documents that specify the communications system and its tests. From them, it proceeds with the Definition of the Test System, which specifies the functionality that it must fulfill. It goes on with the Construction of the Test System and Construction of the Lower Subsystem; these phases have been subdivided into lower level activities. For the Construction of the Lower Subsystem, this division is shown at the bottom of Figure 5.

Although the explanation of the methodology can give the impression that we are working with a single Test Suite, and therefore a single Protocols Module, this is so simply because of the choice made in the use of the language in order to facilitate the description. Considering the use of more than one Test Suite would be like repeating the sequence of activities for each one of them, although the Lower Subsystems would have a certain degree of similarity.

SPICUM servicio de publicaciones

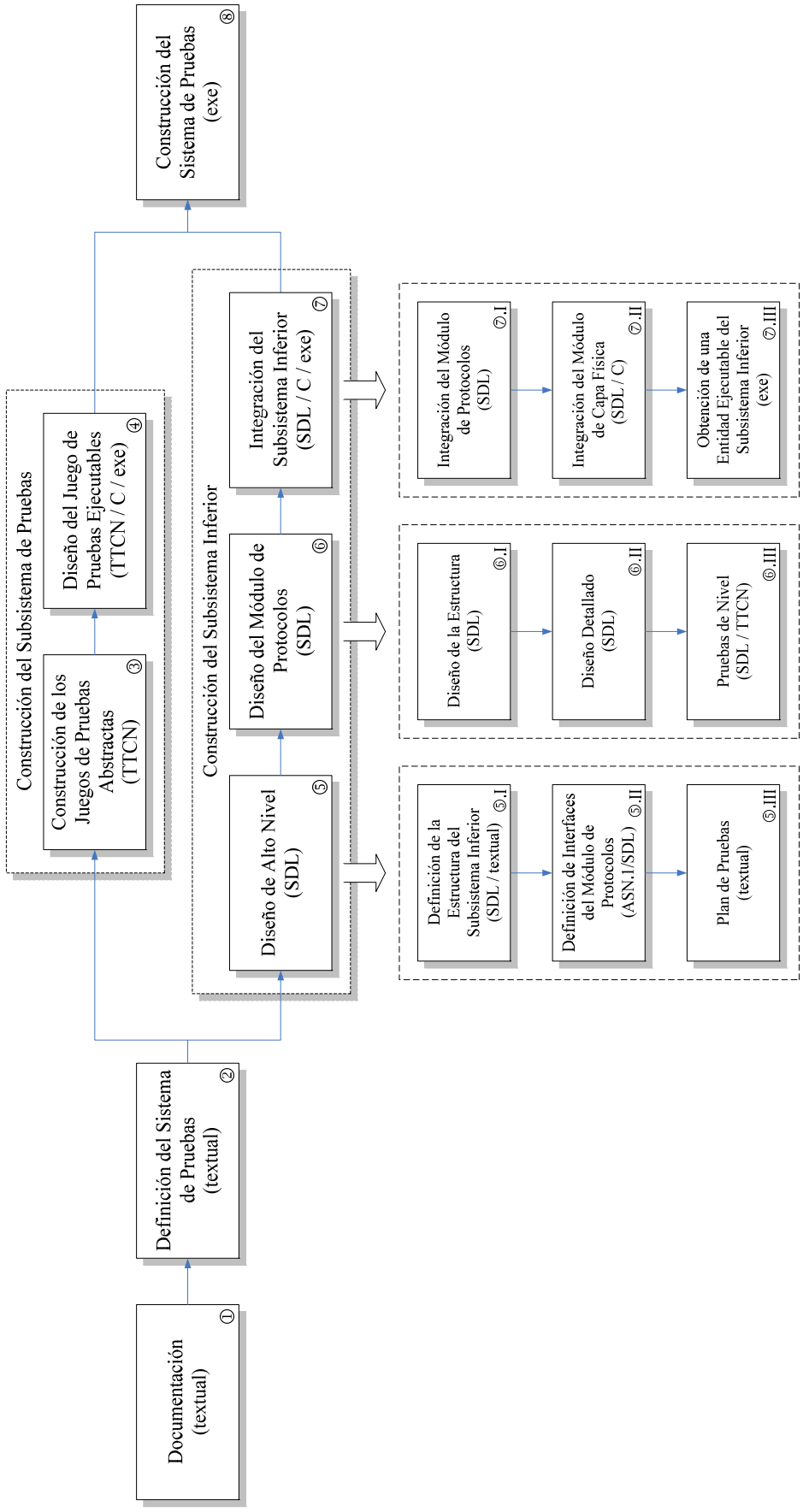


Figure 5: Overview of the Design Methodology.

SPICUM servicio de publicaciones

1.4.2 Definition of the Test System

In the Definition of the Test System phase is selected the functionality that will be implemented the Test System. Choosing the subset of tests implies selecting a specific Test Method.

1.4.3 Construction of the Test Subsystem

The set of steps has been termed 'Construction of the Test Subsystem' includes all activities necessary to obtain an executable for this component of the architecture. It has been divided into two phases:

1. Construction of Abstract Test Suites: Its target is the modeling and validation of the set of Test Cases.
2. Design of the Executable Test Suite: This stage generates an executable of the Test Cases that will be provided by the Test System. This needs implementing some additional elements (TSOs and coders) and linking them with the other components shown in Figure 3.

1.4.4 Construction of the Lower Subsystem

The construction of the Lower Subsystem has been divided into three phases. Given the complexity of these phases each one has in turn been divided into three activities, which appear at the bottom of Figure 5. The responsibilities of each phase are:

1. High Level Design: The target is defining the structure of the Lower Subsystem. Its functionality is split between the Protocols Module and Physical Layer Module, defines the interfaces of the Protocols Module and specifies the Test Plan.
2. Design of the Protocols Module: In this phase, the elements contained in the Protocols Module are modeled. It performs the design of the structure and, then, the detailed design of each element. The phase ends once the Layer Tests previously defined are run on the design.
3. Integration of the Lower Subsystem: Its purpose is the integration, in an incremental way, of the various elements of the Lower Subsystem. The result is an executable entity.

There are several methodologies that consider the full development cycle of systems based on SDL ([OLSE94], [BRÆK93], [REED96] [EKAN95], [BRÆK99], [WITA95]). For the design of the Protocols Module no particular methodology has been chosen, instead its design has been structured as a sequence of activities, in which the existing methodologies agree, and milestones associated with them. In general, these activities are:

- Analysis of requirements: It has been carried out in previous stages.
- Decomposition: Division of system functionality into smaller elements.
- Behavior: Models the behavior of each element.
- Tests: Checking the operation of the model.
- Implementation: Generation of an executable model.

SPICUM servicio de publicaciones

1.4.4.1 High Level Design

This stage generates the first models of the Protocols Module, which will subsequently be refined. The following activities are performed:

a) Definition of the Lower Subsystem Structure.

b) Definition of the Protocols Module Interfaces.

c) Test Plan.

1.4.4.1.1 Definition of the Lower Subsystem Structure

In first place the functional division between the Protocols Module and the physical layer module is decided. The principle that guides this functional division is the time synchronization.

Once done, the internal structure of the Protocols Module is defined. In the Protocols Module, each of the layers it contains is represented as a block of an SDL system. If different Test Suites are used, it may happen that the Protocols Module is different for each of them. In general, their differences consist primarily of the addition or removal of one or more of the layers and perhaps the possible interactions between them. Therefore, it is considered more appropriate to design a single Protocols Module, which allows for the selection of its configuration given the Test Suite to use. This mechanism requires an additional set of signals to select the configuration. The management of these signals is done through an additional block that has been called the Configuration Manager.

In parallel with definition of the structure, the functionality to be provided by each block must be decided. This functionality is divided into:

a) Basic Functionality: It is the functionality gathered in the system specifications, for the level associated with each SDL block.

b) Additional Functionality: It models the special behaviors required by the Test Cases that do not appear in equipment not intended for certification tasks.

The activities undertaken in this subphase are essentially those of analysis and definition, thus the outputs are documents, textual or abstract, of very high level. These include the functional description of the Protocols Module and its block structure.

1.4.4.1.2 Definition of the Protocols Module Interfaces

The Protocols Module has the following interfaces (Figure 6):

a) Interface to the Test Subsystem (upper boundary of the Lower Subsystem): The definition of this interface can be generated automatically from Games the Test Suites. The *GenDef* tool has been built with this purpose.

b) Interfaces internal to the Protocols Module (paths between blocks)

c) Interface with the Physical Layer Module (lower boundary of the Protocols Module): It is a local interface.

d) Additional Control Interface (upper boundary of the Lower Subsystem): It is a non standardized interface used to invoke special behaviors in the Protocols Module or transferring additional information. For example, it can be used during system initialization.

SPICUM servicio de publicaciones

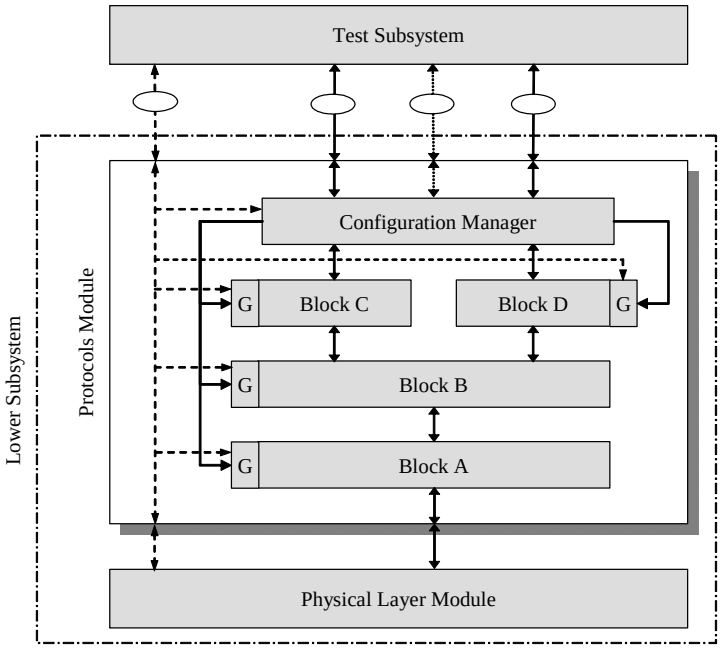


Figure 6: Interfaces of the Protocols Module.

1.4.4.1.3 Test Plan

The Test Plan contains five categories of tests: unit tests, layer tests, modules tests, subsystem tests and system tests.

1.4.4.2 Design of the Protocols Module

The Design of the Protocols Module phase aims to modeling the elements that are part of that module, and takes as starting point the outputs of the previous stage (High Level Design). The language suggested for this modeling is SDL.

This phase has been divided into three activities:

a) Design of the Structure: It defines the processes and procedures that shape the internal structure of the Protocols Module, as well as their functionality and interfaces.

b) Detailed Design: Activity where the behavior of the entities identified during the Design of the Structure is modeled. It involves the construction of the end-to-end coders (Codec $E2E_{SDL}$). Tools can be developed to automatically generate these coders.

c) Layer Tests: The models developed in previous activities are verified here. In order to do so, emulators of the higher and lower layers are built.

The results of this task are SDL models of each layer that forms the Protocols Module, whose behavior has been independently tested.

1.4.4.3 Integration of the Lower Subsystem

The aim of this phase is to integrate the elements that comprise the Lower Subsystem and generating an executable entity for it. Firstly, all layers of the Protocols Module are integrated; then, the Physical Layer Module is integrated. It is suggested that this

SPICUM servicio de publicaciones

integration is carried out in two steps to have a greater control over the errors that may arise. The first step integrates the Physical Layer Module with the lower layer of the Protocols Module and, then, the Physical Layer is integrated with the Protocols Module.

The generation of an executable for the Lower Subsystem is very similar to the generation of an executable for the Test Subsystem. The coder Codec $Local_{TTCN}$ can be generated automatically by the tool *GenCod*.

1.4.5 Construction of the Test System

This last stage targets the integration of the three subsystems that comprise a Test System and the verification of this integration through the system tests. Each subsystem is seen as an executable entity independent of the other subsystems. This phase requires to properly configuring the Operation and Administration Subsystem so that it uses the appropriate combinations of Test Subsystem and Lower Subsystem. The result of this phase is the Test System.

5 Supporting Tools

The tools that have been developed to support the Design Methodology are of two types: generic components of the architecture and automatic generators. Additionally, a set of naming rules have been defined for the SDL models of the Protocols Module. The developed tools are adapted to the Tau Suite development environment.

1.5.1 Generic Components

Within the components of the architecture that are common to all Test Systems the following have been built: the Operation and Administration Subsystem, and the Adapter Module and the Management Module, both for the Test Subsystem and the Lower Subsystem. They are designed to be run on Linux, Unix and Windows platforms.

The Operation and Administration Subsystem has been built in Java [JAVA]. For the design of the graphical interface the Swing library [SWING] has been used. Despite the simplicity of this tool, it has all the functionality needed for an Operation and Administration Subsystem, with the advantage that it can be executed on virtually any platform. The included trace viewer allows analyzing a test execution both in textual and graphical formats. To ease the operator's tasks the tool can filter the traces, selecting which categories to hide. It is also possible viewing the sequence of interactions as a message sequence diagram.

The set of functions that allow the interaction between the Test Suite and the Adapter and Management Modules and in the top boundary of the Test Subsystem follows the standard defined by the CGI interface (Generic Compiler / Interpreter) [GCI96].

The implementation of the components Input/Output Management is based on client-server model. The Test Module Adapter runs as a server for the Operation and Administration Subsystem and as a client for the Lower Subsystem. The interaction with the Operation and Management Subsystem is made through the standard input and output, allowing the use a console interface instead of a graphical interface.

The Time Management component provided by the development environment has been modified to achieve a more accurate resolution. In Windows high resolution hardware counter of the Windows API has been used.

SPICUM servicio de publicaciones

The coder Codec $Local_{TTCN}$ has been designed so it can use both an ASCII transfer syntax (used in the DECT and Bluetooth Test Systems) and a PER one (used in the UMTS Test System).

1.5.2 Automatic Generators

In the group of automatic generators tools have been built to generate the definition of the interface between the Test Subsystem and Lower Subsystem (*GenDef*) and generate the appropriate encoding and decoding functions (*GenCod*). It has also been built an automatic generator for messages exchanged between DECT entities.

The Test Suites, seen as substitutes for one or more layers of a communications system, use two types of interfaces. A local interface with the adjacent levels, internal to the Test System, and an end-to-end interface comprised of the messages exchanged between the Test Case and the Implementation Under Test. We have worked in the automatization of both types of interfaces, and we have designed a Generator of Local Interfaces (*GenInt*) and a Generator of the Coder at the Air Interface (*GenCodecAir*).

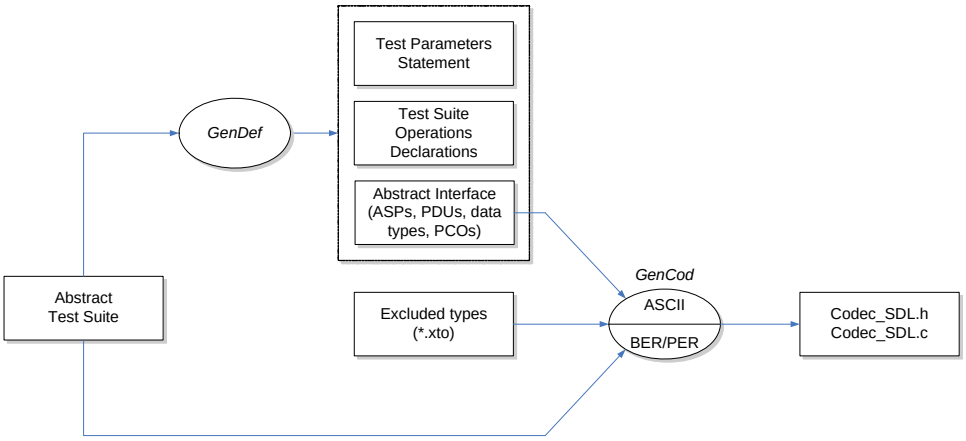


Figure 7: Schematic diagram depicting the use of the Generator of Local Interfaces.

The Generator of Local Interfaces (*GenInt*) (Figure 7) allows to automatically implementing the interface between the Test Subsystem and the Lower Subsystem. It consists of two tools: the Generator of the Definition of the interface (*GenDef*), which extracts the set of data types used in the interface by the Test Subsystem, and the Generator of the Coder for the interface (*GenCod*), which automates the construction of the corresponding encoders and decoders for both components.

These tools have evolved at the same pace than the Test Suites and the commercial development environments. Therefore, initially they made use of an ASCII transfer syntax, and only accepted TTCN data types; their final implementation uses a BER/PER transfer syntax.

The *GenDef* tool processes the `MP` format of the Test Suites and generates a set of files containing the definitions of primitives (ASPs), data units (PDUs) and data types used in the interface between the Test Subsystem and the Lower Subsystem. The tool processes all the possible restrictions such as maximum value, value ranges, etc.

For its construction the analyzer generator PRECCX [BREU97] has been used. The TTCN grammar [X.292] has been preprocessed in order to adapt it to PRECCX. Some

SPICUM servicio de publicaciones

rules have been modified to suit the appropriate characteristics of PRECCX such as, for example, those which requires that the longest sequence precedes the others (rule 680), the recursion on the right-side is better than the left-side (rule 707), etc. The augmented grammar is constructed including, in C language, the desired action on the rules of the grammar. The *GenDef* tool has been developed and initially used for the development of Test Systems for DECT.

From the outputs generated by *GenDef*, the *GenCod* tool allows to generate, automatically, the coder component of the Lower Subsystem. This coder has been implemented in C, as this language allows an easier handling of the information than SDL. The tool has been adapted to the characteristics of the code generators of the Tau Suite development environment. The code generator that owns this environment for TTCN allows the use of two transfer syntaxes (to communicate with the Lower Subsystem): ASCII and BER/PER. The *GenCod* generator has a version for each transfer syntax.

The third tool that has been designed is the Generator of the Coder at the Air Interface (*GenCodecAir*) [PONC00], which automatically generates the procedures for coding and decoding end-to-end messages for the Link and Network layers. The tool is adapted to the transfer syntax of the DECT system ([ETS 300 175-4], [ETS 300 175-5]). This generator uses as inputs the files that contain the data type definitions previously created by *GenDef*. The result is a file in textual format (PR), which can be used in an SDL model. For each PDU and data type is created a procedure for encoding and another for decoding. The use of macros helps to simplify the generated code.

6 Architecture for Radio Test Systems

Traditionally, conformance testing and radio protocols have been considered distant worlds, as engineers who work in them often have no training in both fields. While protocol tests are targeted to finding that the sequences of messages exchanged between two entities are performed in the correct order and with the proper syntax, the goal of radio tests is to certify compliance with aspects such as radioelectric compatibility in transmission and reception.

One essential difference is that radio conformance tests require specific instrumentation (signal generators, modulators, oscilloscopes, spectrum analyzers, etc..) which can carry out a certain set of measures over the air interface. However, most of the concepts are shared between the two types of tests. Radio tests are also standardized using a similar methodology, with the only exception that the testing methodology only gets to describe the scenarios and the expected behavior of the equipment at the level of natural language.

We propose a methodology ([PONC07a], [PONC07b], [GOME01a]) for the design of radio Test Systems that brings this field to the level reached by the protocol testing. It is based on the assimilation between the protocol tests and the radio tests, seeing the latter as a sequence of messages exchanged between the Test Case and the Instrumentation. The formalization of radio tests includes modeling these interactions through an interface of abstract primitives in TTCN, which allows the communication with the Lower Subsystem.

It is shown that it is possible to design Test Systems using the same architecture as for protocol Test Systems and to use all the tools already available in the area of protocol testing. This architecture allows the integration of instruments from different

SPICUM servicio de publicaciones

manufacturers as well as the immediate replacement of any equipment with another of similar capabilities.

1.6.1 Overview of Radio Test Systems

Radio tests are electrical tests of the air interface, where it is determined whether the EUT (Equipment Under Test) transmits and receives within the limits (frequency, power, ...) set in the System Specifications. From a conceptual point of view a radioelectric measurement (voltage, current, frequency, ...) in a radio test is equivalent to the interpretation of the sequence of bits that form one frame in a protocol test.

A radio Test System is composed of the elements shown in Figure 8. The operator interface is equivalent to that provided in protocol Test Systems, but must include a module for the graphic representation of radio measurements. The Measurement System is responsible for the test realization and for carrying out the electrical measurements. The Measurement System basically consists of instrumentation, remotely controlled, and a control module, which coordinates the realization of the measurements and the creation of the radioelectric scenario. The connections between the different instruments depend on the test that is run. These are realized through a switching matrix. Finally, the Signaling Unit is responsible for taking the EUT to the state required for each test; this element is directed by the control module of the Measurement System.

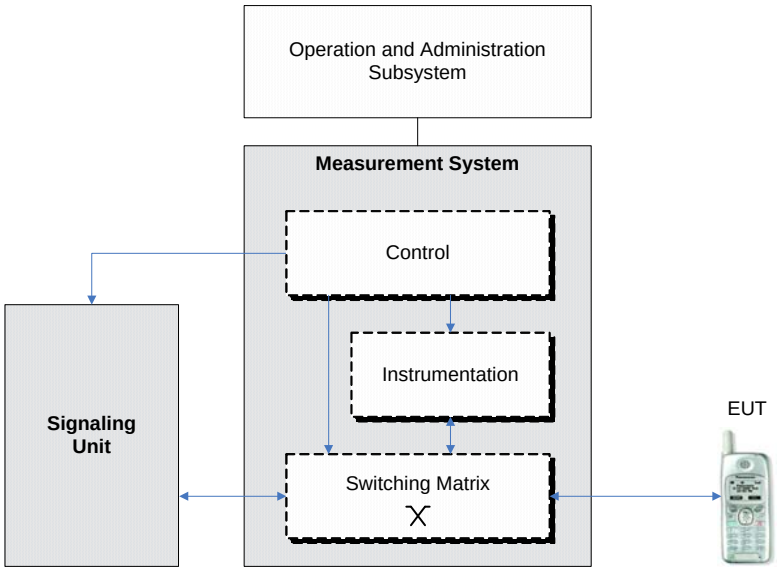


Figure 8: Elements of a Radio Test System.

To overcome the current limitations we propose to use the following principles as an integral part of the design process for these Test Systems:

- Use the same architecture as for protocol Test Systems.
- Formalize radio tests using the same notation as for protocol tests.
- Use the same development environments.

Overall, being able to reuse experience, tools and languages, it is achieved a cost reduction and an improvement of the design process.

SPICUM servicio de publicaciones

1.6.2 Architecture of Radio Test Systems

Figure 9 shows how it is possible to adapt the architecture of protocol Test Systems to radio Test Systems.

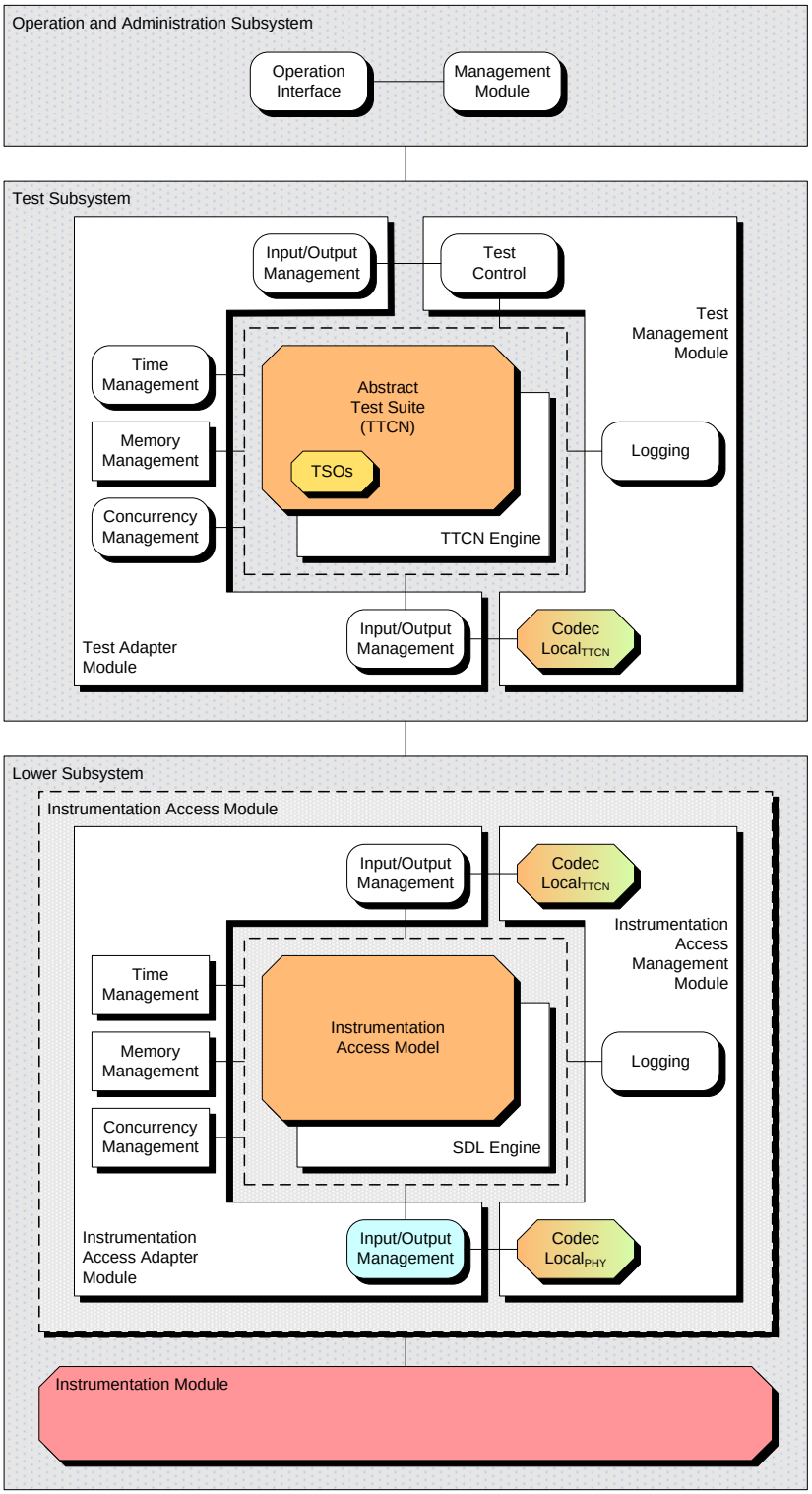


Figure 9: Specific components of a radio Test System.

The Measurement System includes the Test Subsystem and the Lower Subsystem:

SPICUM servicio de publicaciones

- The Test Subsystem controls both the Signaling Unit and the Instrumentation.
- The Lower Subsystem provides access to the air interface of the EUT. Within it, the Instrumentation Module encompasses the measurement instrumentation plus the switching matrix, and the Instrumentation Access Module is responsible for adapting the communication between the Instrumentation and Test Subsystem.

It is possible to build a codec, Codec $Local_{PHY}$, reusable between different radio Test Systems. The TTCN codec, Codec $Local_{TTCN}$, needs to be adapted to the set of primitives and messages used in the interface between the Test Subsystem and the Lower Subsystem.

The Signaling Unit is an element that can be considered an independent entity in itself. It can be constructed using the protocol Test Suites and their corresponding Lower Subsystem. The protocol Test Cases already contain the test steps necessary to place the EUT in most, if not all, of the initial states for the radio tests. The superfluous code can be removed in order to extract the initialization and termination sequences that are needed.

1.6.3 Radio Abstract Test Suites

A radio Test Case is divided in the same phases as protocol Test Cases (preamble, body, and postamble). Radio Test Cases have a more lineal appearance, because, once the EUT is placed in the desired state, the measure is performed and the test finishes; possible alternative branches of communication with the EUT are hidden in the Signaling Unit. In a radio Test Case, seen at high level, the actions shown in Figure 10 are performed. The signal processing that a radio Test Case may require is modeled in external functions, which are called from the Test Cases.

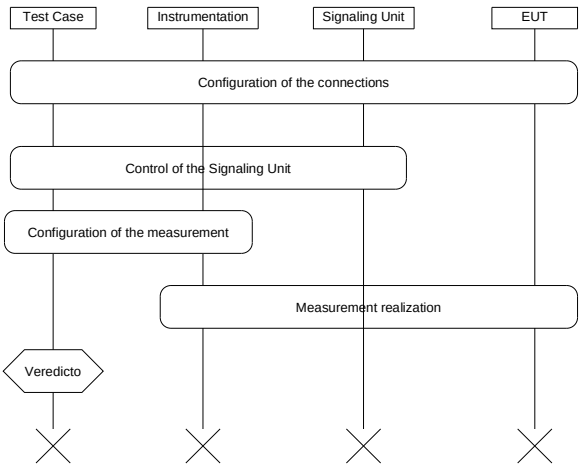


Figure 10: Typical sequence of actions in a radio Test Case[1].

One measurement is characterized by:

- Measurement type

[1] When the line of one entity crosses a boxes on top of it, it shows that the entity that not intervene in that activity.

SPICUM servicio de publicaciones

- Parameters associated with the measurement type and
- Interconnection (electrical configuration) of the Test System elements.

Since the test cases are abstract, in order to faithfully fulfill the Test Purpose it is necessary to abstractly model the concept of measurement. This abstraction implies two aspects: the abstract definition of the instrumentation required for such measurement, specifying both its semantics and its electrical configuration, and the communication with the other elements of the Test System, in order to set the parameters of the measurement to perform.

1.6.3.1 Modeling of the Instrumentation

To build an abstract model of the Instrumentation we define the concept of Virtual Instrument (IV). A Virtual Instrument is an element which can perform one or several measurements, each of which can be configured with zero or more parameters, with one or more input and/or output connectors. The switching matrix can be viewed as a set of electrical paths, each of which can be modeled as a Virtual Instrument.

To model a virtual instrument the following characteristics must be specified:

1. Electrical ports (connections) and direction (input or output).
2. Types of measurements that can perform (semantics).
3. Configurable parameters for each type of measurement.

The definition can be made in the same document that specifies the Test Method, either by a complete specification of each Virtual Instrument or referencing some other standard where such Virtual Instruments have been specified. This definition could be done in natural language. A Virtual Instrument can be, for example, a power meter, and its configurable parameters might be the frequency to be measured, the type of measurement (average power, peak power, ...), the duration of the measurement, and so on. The most satisfactory option to model these instruments would be within the TTCN module, but the semantics of TTCN does not allow it because it is not possible to declare elements which are external to the Test Suite.

1.6.3.2 Interface with the Instrumentation

The interface [PONC07a] which allows controlling the Instrumentation from the Test Suite consists of five primitives (Table 2). All of them are confirmed.

1.6.3.3 Implementation of the Instrumentation Access Module

The Instrumentation Access Module is responsible for:

- Managing the physical interface with the Instrumentation, adapting the interface expected by the Test Subsystem.
- Translating the commands sent by the Test Subsystem into commands specific of the Instrumentation and representing them in the proper format.
- Hiding differences between instruments from different suppliers.

The design we have made allows the integration in the Test System of instrumentation from different suppliers, but also takes into account the fact that a command may have different meanings for each equipment. Access to the Instrumentation has been realized

SPICUM servicio de publicaciones

via the GPIB bus [IEEE 488]. The local state of the GPIB bus is considered the idle state. To control the instrumentation we have used the standard SCPI commands (Standard Commands for Programmable Instrumentation) [SCPI99].

Table 2: Primitives for communication with the Instrumentation Access Module.

Primitive	Parameters
INIT_INSTRUMENT_REQ	(INSTRUMENT id, ListPORTS lp)
INIT_INSTRUMENT_RSP	(INTEGER cod_error, IA5String cad_error)
CONNECT_REQ	(INSTRUMENT id1, PORT port1, INSTRUMENT id2, PORT port2)
CONNECT_RSP	(INTEGER cod_error, IA5String cad_error)
SET_PARAMETER_REQ	(INSTRUMENT id, INSTPAR par, PARVAL val
SET_PARAMETER_RSP	(INTEGER cod_error, IA5String cad_error)
GET_PARAMETER_REQ	(INSTRUMENT id, INSTCOM com, COMVAL val)
GET_PARAMETER_RSP	(IA5String measure, INTEGER cod_error, IA5String cad_error)
STOP_INSTRUMENT_REQ	(INSTRUMENT id)
STOP_INSTRUMENT_RSP	(INTEGER cod_error, IA5String cad_error)

The adaptation of the Instrumentation Access Module for a specific instrumentation is done via two types of configuration files. The first one shows the available equipment and the information needed for their addressing. Each instrument has an additional file where commands, and their parameters, used in the Test Suites are associates with the corresponding commands, and their parameters, of the instrument. If a Test Case command requires more than one Instrumentation command, these are executed sequentially. When all commands have been executed, a response is returned confirming the operation.

7 System Design with SDL

There exists a wide range of methodologies for object-oriented design [BURK96]. Among them we can mention the methodologies of Booch ([BOOC94], [BOOC95]), Coad/Yourdon [COAD91], Coleman [COLE94], Jacobson [JACO92] and Rumbaugh [RUMB91]. For designs based on the SDL language more specific methodologies have been proposed. Some of them are SPECS ([SPEC93], [OLSE94]), SDL+ [REED96], SOMT [EKAN95] and TIMe [BRÆK99]. Other proposals are [BORN98] or [VERI96].

This chapter describes the analysis made on the SOMT methodology. As a result of this study amendments to the methodology are proposed in order to adapting it to design processes based on standards. The resulting methodology is called M-SOMT (Modified SOMT) [ALBA00a]. For this evaluation we have implemented the DECT Network Layer [ETS 300 175-5] following the steps defined in this methodology.

The SOMT methodology (SDL-oriented Object Modeling Technique) suggests a set of activities or phases, and an order among them, to carry out the design and implementation of systems under the object-oriented paradigm adapted to the SDL language. This is an adaptation of the OMT methodology [RUMB91] but also incorporates influences from UML and other methods of analysis such as Jacobson. Examples of use of the SOMT methodology are [KEUM98], [YEWE00] and [RODR07].

The methodology defines five activities: Requirements analysis, System analysis, System design, Object design and Implementation. Figure 11 shows the activities that

SPICUM servicio de publicaciones

constitute the methodology (left), along with the models that are generated in each (right). The main models of each activity and the relationships between them are shown in the central part of the figure.

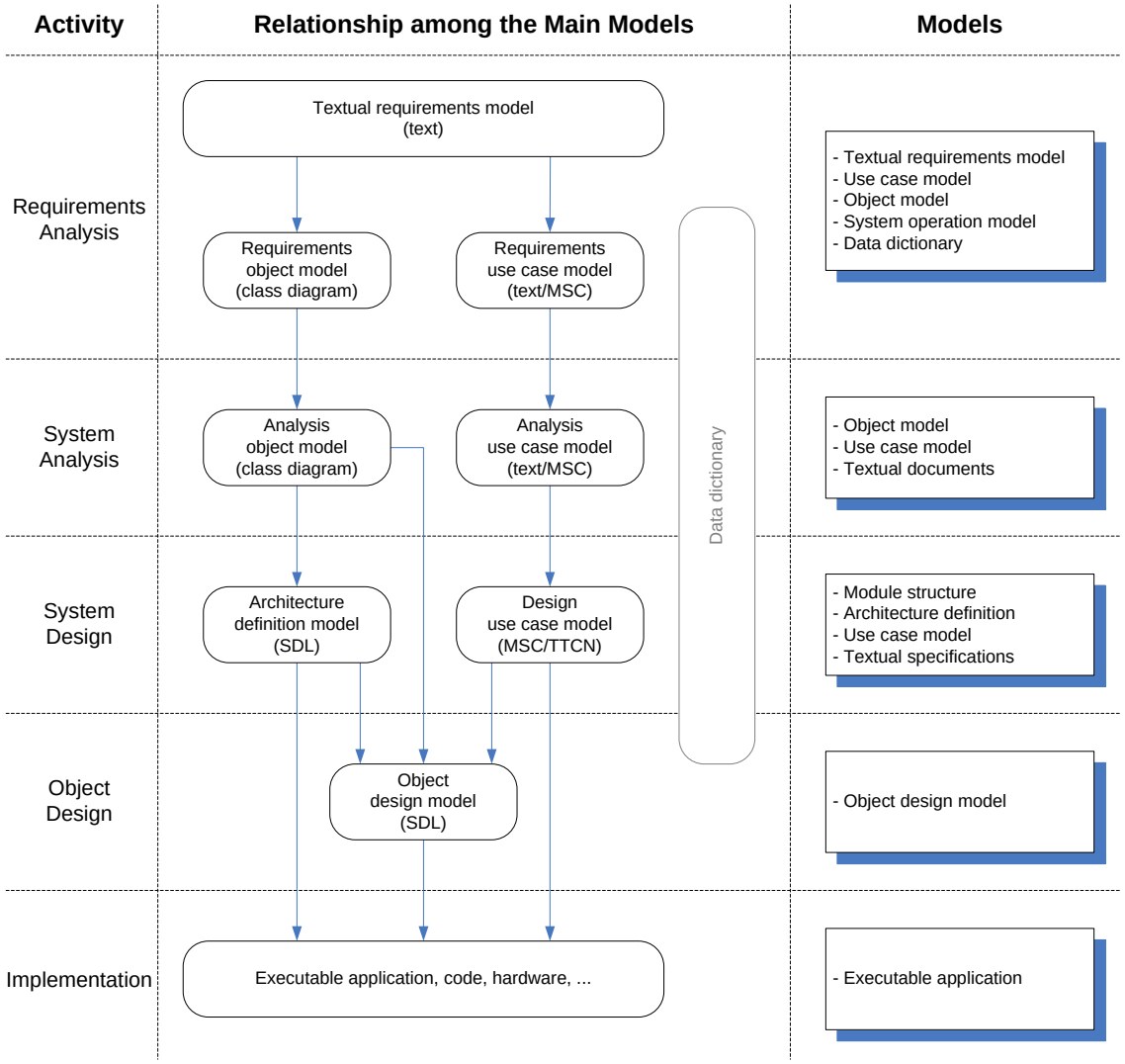


Figure 11: Activities and models of the SOMT methodology.

1.7.1 Modified SOMT Methodology (M-SOMT)

For designs based in standards, the requirements analysis activity ceases to have effect, as the study of the problem domain and the constraints it imposes on the system has been made during the development of the standard. The standard is, thus, incorporated as a document on which later models will be based.

The System analysis activity has been kept unchanged. It develops the analysis use case model, which presents an overview of the behavior of the system entities and the analysis object model, which provides a first outline of what the structure of the system will be.

In the System design activity many changes have been introduced. Not only models that have been deemed irrelevant have been removed, but it has also incorporated the design

SPICUM servicio de publicaciones

object model. Both this model and the design module structure are expressed on the notation used for the analysis object model. The architecture definition model already uses the language that will be used in the final abstract model, SDL. This definition is an intermediate step in the elaboration of the final model.

It has been decided that, unlike the SOMT methodology, the definition of architecture includes the definition of all elements derived from the design object model, which encompasses not only the blocks, as originally intended, but also the processes and specialized process type classes present in the model. The methodology suggests several possible structures for each object type, according to its function, but it is not possible to automate the transition between both models, as this transformation is subjected to numerous design decisions.

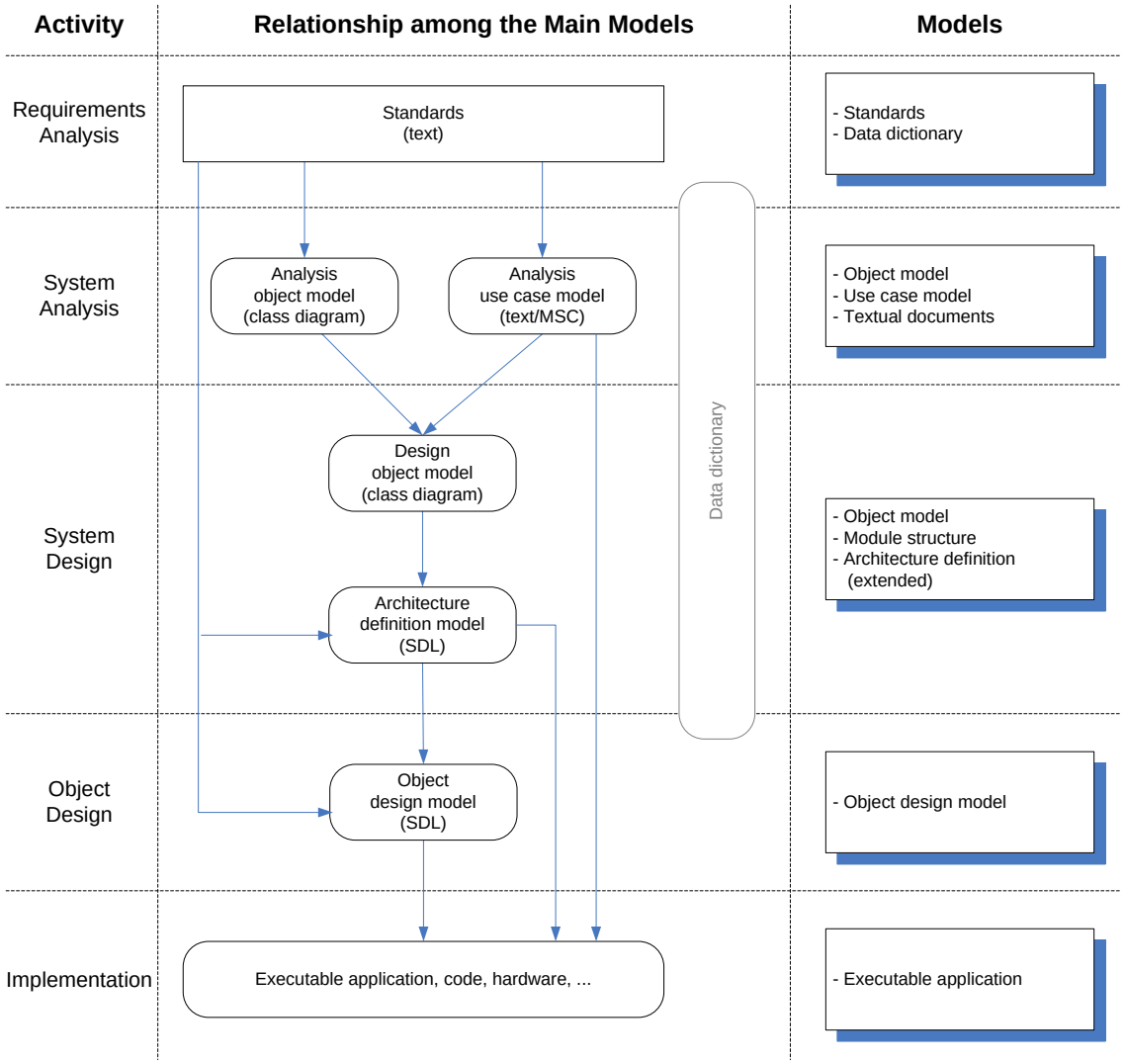


Figure 12: Activities and models of the M-SOMT methodology.

The Object design activity consists of the description in SDL of the behavior of each element of the architecture. During this modeling, new procedures will appear and will be incorporated into the architecture. The result is an SDL model which offers the functionality demanded by the Requirements analysis and meets the imposed restrictions.

SPICUM servicio de publicaciones

The fifth and last activity, the Implementation, depends largely on the characteristics of the final platform. The original methodology only provides some very general indications. Therefore, it is deemed appropriate for its description. In the implementation that uses modeled Network Layer, in general, tasks were performed in the suggested order, as seen in Chapter 8 of the Thesis.

The set of models and activities of the M-SOMT methodology is shown in Figure 12. The methodology presents the advantages associated with the use of the object-oriented approach as well as those which arise from the use of formal languages.

8 DECT Test Systems

This chapter presents a detailed description of the application of the Design Methodology to the construction of Test Systems for DECT DLC and NWK Layers (Digital Enhanced Cordless Telecommunications) [ETS 300 175]. This work has been carried out under a project funded by the European Union [PROY99]. The implementation of these Test Systems validates the Design Methodology presented in Chapter 4. The tests that have been implemented are those standardized for the GAP profile [ETS 300 176-2], which provides the basic functionality for voice telephony applications.

This explanation is structure in the same order as the description of the Design Methodology to provide a better view of the effort involved and the results achieved in each stage and facilitate their comparison with the presentation of the methodology. The description of each activity has been made with a high degree of detail to show a precise vision of the application of the methodology to a real case study.

DECT is a digital short-range wireless telephony system deployed around the world [ETR 183]; it is part of the ITU IMT-2000 family [PEJA02]. DECT has been designed to operate in public, residential or business environments. The communication is performed from a Mobile Termination (PT) (mobile) via a Fixed Termination (FT) (base station) that acts as a gateway to the public telephone and data networks (GSM, X.25, ISDN, UMTS). The most important characteristics in which this technology is based are the dynamic channel selection, the use of unlicensed spectrum, the ability to move between different areas of coverage, and the security it provides to communications. DECT has become the dominant technology in the wireless market and in the residential segment of the PABX (Private Automatic Branch Exchange) [DECT], being used in several applications ([MORE95], [BERW96], [REIS95], [WLL600], [NATI01], [PRIX02]).

1.8.1 Documentation

The Test Structure And Purposes for DECT protocols tests are defined as separate parts for each protocol layer; as a whole it is called the Test Case Library (TCL) [EN 300 497]. For GAP profile, the Profile Test Specification (PTS) is described in Part 1 of document [ETS 300 494]. The Abstract Test Suites only support the embedded remote single layer Test Method.

1.8.2 Construction of the Test System

The Executable Test Suites (ETS) are generated from the Abstract Test Suites published by ETSI. It has been necessary to slightly go over them as some aspects were not

SPICUM servicio de publicaciones

defined according to the System Specifications. For example, although standard [ETS 300 175-4] defines some primitives with optional fields, or even omitted, the TTCN code required the presence of those fields. Changes have been made in these cases substituting the specification of reception '?' (expects to receive anything) by a wildcard '*' (expects to receive anything or none). The Test Parameters files and the TSO function declarations were generated from the Abstract Test Suites using the GenDef tool.

1.8.3 Construction of the Lower Subsystem

1.8.3.1 High Level Design

The separation between the Protocols Module and the Physical Layer Module has been realized at the Medium Access Layer. To implement the Physical Layer Module, commercial boards [SIT94a] have been chosen; these boards implement the Physical Layer and lower part of the MAC Layer (CSF - Cell Site Functions). Table 3 lists the subcomponents within each module. Figure 13 shows the structure of the Protocols Module for the Portable Termination Network Test System.

Table 3: Subcomponents included in each Module of the Lower Subsystem for each Test System.

		Test Systems		
Module		**SP_DLC_PT**	**SP_DLC_FT**	**SP_NWK_PT**
Protocol Modules	*Name*	*MProt_DLC_PT*	*MProt_DLC_FT*	*MProt_NWK_PT*
	Blocks	LLME_MAC_FT SUB_DLC_FT MAC_CCF_FT LINSER_FT	LLME_MAC_PT SUB_DLC_PT MAC_CCF_PT LINSER_PT	LLME_FT DLC_FT MAC_CCF_FT LINSER_FT AJUSTE_TIPOS_FT
Physical Layer Module	*Name*	*MFis_FT*	*MFis_PT*	*MFis_FT*
	Blocks	MAC_CSF_FT PHY_FT	MAC_CSF_PT PHY_PT	MAC_CSF_FT PHY_FT

The interface with the Test Subsystem has been obtained by applying tool GenDef to the Test Suites. The control interfaces have been realized via the management block, LLME. The signals of these interfaces are grouped into those used by the TSO functions and those used for the control of the Lower Subsystem. The signals used in each interface are described in detail in the Thesis.

Throughout the design process the generated models are verified with the tests specified in the Test Plan defined at this point. Until the Physical Layer Module is integrated the simulation tools provided by the development environment are used. As layer tests both our own tests and the ETSI Abstract Test Suites have been used. For the Protocols Module testing, the structure shown in Figure 14 has been used. As subsystem and systems tests, the corresponding Test Suites have been used.

SPICUM servicio de publicaciones

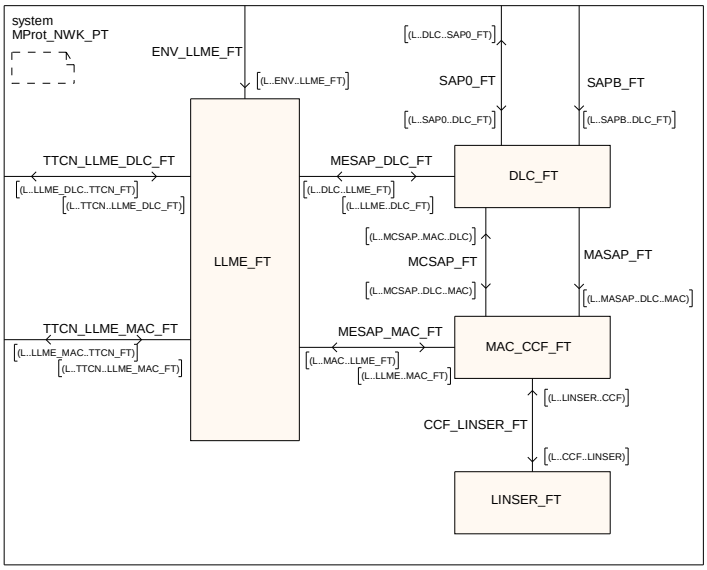


Figure 13: Structure of the Protocols Module for the Portable Termination NWK Test System.

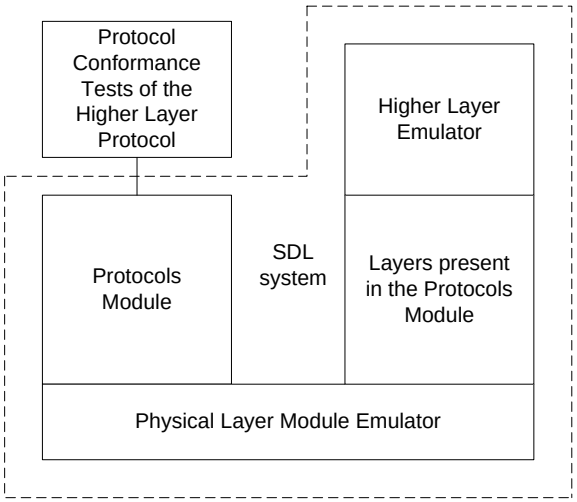


Figure 14: Generic structure of the Module Tests.

1.8.3.2 Design of the Protocols Module

The design of the internal structure of each block has attempted to follow the structure suggested by the standard that describes its behavior. To properly organize the design, a set of auxiliary packages that contain elements used by the blocks presented in the preceding paragraphs have been defined. Each Test System incorporates a different set of blocks, although some of these blocks are used in more than one Test System. This information is summarized in Table 4. The description of the models of the processes has been included in Appendix J of the Thesis, which shows the states and transitions that each process owns and their behavior. For the layer tests, emulators of the Network and Link Layers both for the Portable and Fixed Terminations and of the combined PHY and MAC Layers have been built.

SPICUM servicio de publicaciones

Table 4: Processes contained in the blocks included in each Test System.

			Test System		
Layer	**Blocks**	**Processes**	**SP_DLC_PT**	**SP_DLC_FT**	**SP_NWK_PT**
Medium Access	MAC_CCF_FT	BMC	✓	-	✓
		MBC_CTRL			
		MBC			
		MBC_SELEC			
	MAC_CCF_PT	BMC	-	✓	-
		MBC_CTRL			
		MBC			
		MBC_SELEC			
	LINSER_FT	LINSER_FT	✓	-	✓
	LINSER_PT	LINSER_PT	-	✓	-
Link Control	DLC_FT	CTRL_FT	-	-	✓
		LAPC_FT			
		Lc_FT			
		SignalROUTER			
		Lb_FT			
	SUB_DLC_FT	ConversorTTCN	✓	-	-
		Cuasi_Lc			
		Signal_RTX			
	SUB_DLC_PT	ConversorTTCN	-	✓	-
		Cuasi_Lc			
		Signal_RTX			
	AJUSTE_TIPOS_FT	AJUSTE_TIPOS_FT	-	-	✓
Management	LLME_FT	LLME_DLC_PT	-	-	✓
		LLME_MAC_PT			
	LLME_MAC_FT	LLME_MAC_FT	✓	-	-
	LLME_MAC_PT	LLME_MAC_PT	-	✓	-

1.8.3.3 Integration of the Lower Subsystem

The integration of the Protocols Module of each Test System has been tested via an SDL system with contains an emulator of the System Under Test; the communication between both is carried out via an emulator of the Physical Layer Module.

The integration of the Protocols Module with the Physical Layer Module has been realized via a driver developed in C. The codec used in the communication with the Lower Subsystem, Codec $Local_{TTCN}$, has been automatically generated with tool GenCod and uses ASCII transfer syntax. Several System Under Test emulators have been built to verify the behavior of each integrated Lower Subsystem. In these tests the Protocols Module has been executed as a platform application; the corresponding System Under Test emulator has been executed within the simulator using the real time engine.

1.8.4 Construction of the Test System

The Test Systems have been built and tested on Linux, Unix and Windows platforms. In addition, the DLC Test System for the Portable Termination has also been ported to a platform with the DSP/BIOS operating system [TIBIOS] to demonstrate the use of real-time platforms and the flexibility of the architecture [PLAZ06]. The allocation of components is shown in Figure 15. The code generated from the SDL model has been integrated with the operating system using a light integration model [STAM97].

SPICUM servicio de publicaciones

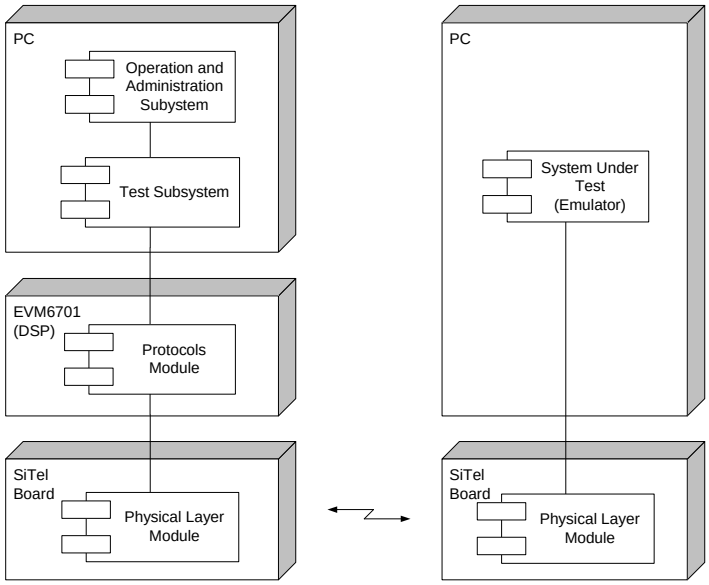


Figure 15: Allocation of components when a real-time platform is used.

The Protocols Module is executed as two tasks. The main task is responsible for executing the SDL system, including the scheduling of SDL processes and the communication between them, and sending messages towards the Test Subsystem. The second task has lower priority and is responsible for receiving messages from the Test Subsystem. The Protocol Adapter Module has required slight modifications of its components to adapting it to the DSP/BIOS operating system. In the Test Subsystem, the Test Adapter Module has had to be adapted to realize the communication with the DSP.

9 Bluetooth Test Systems

The experience gained during the design of the DECT Test Systems has been applied to design of a Bluetooth protocol Test System within the collaboration ([PROY02], [CONT99b], [CONT02]) with company AT4 wireless [AT4W]. This application has allowed enhancing the tools and components used with the suggestions of the company design group. The resulting Test System, BITE (Bluetooth Qualification Tester), has been industrially exploited, becoming the most sold Bluetooth protocol conformance Test System throughout the world [BITE].

In this chapter are presented the distinguishing aspects in the design of the Bluetooth Test System in comparison to the design described in the previous chapter. The Bluetooth technology has also been the area in which the flexibility of the methodology has been checked through the use of different development environments and the application of the architecture to the design of interoperability Test Systems. Test Systems have been built for LM and SPP layers [SORE03a] together with an interoperability Test System for the Headset profile ([MORI03], [MORI02a]).

The architecture of a Bluetooth system is shown in Figure 16.

SPICUM servicio de publicaciones

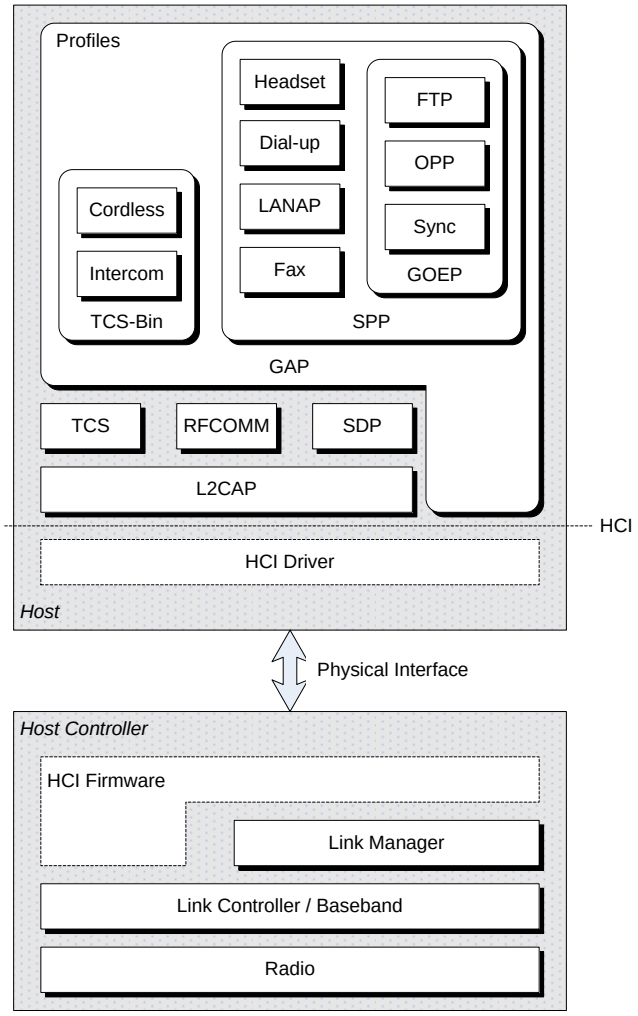


Figure 16: Bluetooth architecture.

1.9.1 Conformance Test System

We have designed the Protocols Module and, both in the Test and Lower Subsystems, the developed Adapter and Management Modules as well as the tools built for the methodology have been used. The communication between the Test and Lower Subsystems use the ASCII transfer syntax; tool GenInt has being used to generate the Codec $Local_{TTCN}$ of the Lower Subsystem.

The structure of the Protocols Module is shown in Figure 17. It consists of 3 processes that implement the L2CAP layer, the HCI interface and a manager that starts and can terminate the operation of both. The implementation has been verified with the SIG tests.

The integration between the L2CAP and HCI has been verified, first, through emulation of the layers below the HCI. Then, the Protocols Module has been integrated with a Physical Layer Module from Ericsson (EBDK - Ericsson Bluetooth Development Kit) [EBDK01].

This Protocols Module have been later completed with implementations in SDL of SDP, RFCOMM, SPP and HID, and in C of GAP and OBEX ([VIGO04], [ROME03], [TERN03], [LARA04]). And multiple applications have been developed above them ([FERN04],

SPICUM servicio de publicaciones

[GIMI05], [ESPI05], [SERR06]). Also, a USB driver has been built [CARD05] to use commercial devices with the developed but Bluetooth stack, but it has not been integrated in the Test System.

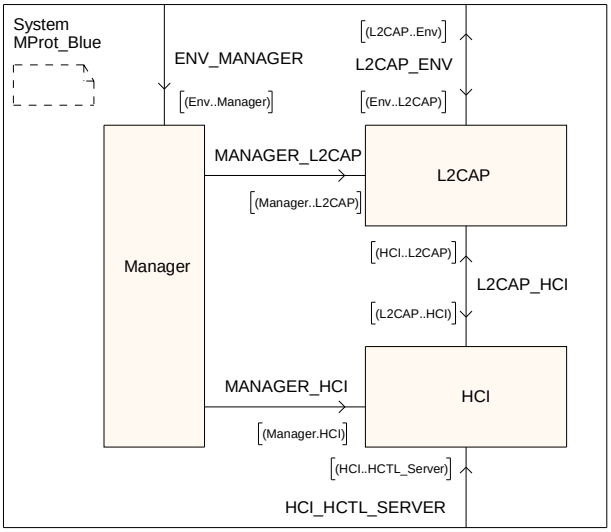


Figure 17: Structure of the Protocols Module for Bluetooth.

The Test System includes improvements in some of the components with respect to those used in the DECT Test Systems. In the Test Subsystem, the Concurrency Management component has been included, which was not previously used. This component creates a thread for each Test Component and offers a service of semaphore protected tasks. The Input/Output Management component of the Lower Subsystem has been slightly modified, and includes the capability to informative traces of the Protocols Module execution towards the Operation and Administration Subsystem. The organization of the automatically generated code has also been improved.

1.9.2 Use of an Alternative Commercial Tool

The use of tool TTCN Toolbox [DANET] has been evaluated as an alternative for the generation of the Test Subsystem. Using this tool, Test Systems for the SPP and LM layers have been built ([SORE03a], [SANC02]). The Serial Port Profile (SPP) [BT SPP] defines the requirements for establishing an emulated serial port between two Bluetooth devices using RFCOMM [BT RFCOMM]. Its Test Suite includes 29 Test Cases. The Link Manager layer (LM) [BT LM] is primarily responsible for establishing and controlling the link and for the power management. Its Test Suite includes 104 Test Cases [BTEST LM].

Due to using a different tool, the interface of the Test Subsystem generated by the new tool with the Lower Subsystem has had to be adapted. Toolbox TTCN uses, in the interface with the Lower Subsystem, a binary transfer syntax based on the BER encoding rules. However, the syntax is ambiguous since it is not able to identify the missing fields when it encodes a value. For this reason, the encoding and decoding functions created by TTCN Toolbox have been generated externally [SORE02]. Tool GenCod has been modified so that it generates these functions, keeping the definition, from the data types extracted by tool GenDef from the Test Suite. In addition, on the

SPICUM servicio de publicaciones

provided libraries the following changes have been made: use of the Little Endian format for strings of bytes and bits, differentiated treatment of ASPs and PDUs, and use direct encoding for all data types.

The following Physical Layer Modules have been used: a device CSR (Cambridge Silicon Radio) [CSR02], EBDK Development Kit (Ericsson Bluetooth Development Kit) [EBDK01] and the Physical Layer Module included in the BITE Test System. For the system tests, as IUT an Axis OpenBT stack on a Windows platform has been used.

As a result of this evaluation it can be said that essentially both tools (Tau TTCN Suite and Toolbox) possess a similar complexity, although difficulties found have been different in each. As advantage, the evaluated tool provides a greater variety and flexibility in the coders of the interface with the Lower Subsystem.

1.9.3 Interoperability Test Systems

To demonstrate that it is possible to use the architecture proposed by the methodology for both conformance and interoperability Test Systems, a prototype of interoperability Test System for the Headset profile has been designed [MORI03]. The Headset profile [BT HEAD] allows users to establish audio connections by using hands free devices. This profile uses protocols L2CAP, RFCOMM and SDP; signaling is realized via AT commands [ATCOM]. It has 34 different Test Cases.

The Test Subsystem has been developed in C [MORI02a], since one of the objectives was that the user of the Test System could design its own Test Cases, modifying the provided ones or from scratch. A function has been built for each Test Case. The Protocols Module has been implemented with the Axis OpenBT stack [AXIS01] above which it is incorporated the functionality of the Headset profile. The Axis OpenBT stack v0.0.8 has been ported to Windows, because the distribution only ran on Linux.

10 UMTS Test Systems

The Design Methodology has been applied to the construction of a commercial protocol conformance Test System for UMTS (MINT - Mobile Communications Integrated Tester) ([PROY02], [PROY03], [CONT02], [CONT03a]). In particular, there has been a direct involvement in the design of the Protocols Module ([COLA02], [COLA03], [SORE03b]). The participation in the design of this conformance Test System has been one more step that has helped evolve the Design Methodology and the associated tools. Furthermore, progress has been made in formalizing the design of interoperability Test Systems, integrating commercial tools in the process and adapting our own tools [MORI04]. The result of the collaboration has also been used as Signaling Unit in the radio conformance Test System.

The philosophy in which the design of the Test Suites standardized by 3GPP is based has evolved in some aspects. One of these points is the Test Methods specified for the various protocols; a unique architecture has been defined for all Test Cases, with the possibility to use additional modules when needed. Therefore, the structure of the Protocols Module can dynamically select the internal configuration depending on the tests to be performed. Furthermore, ASN.1 has been adopted as the preferred.1 notation for primitives and data units. This has led to extend the automatic interface generators already available including the PER transfer syntax at the interface between the Test Subsystem and the Lower Subsystem.

SPICUM servicio de publicaciones

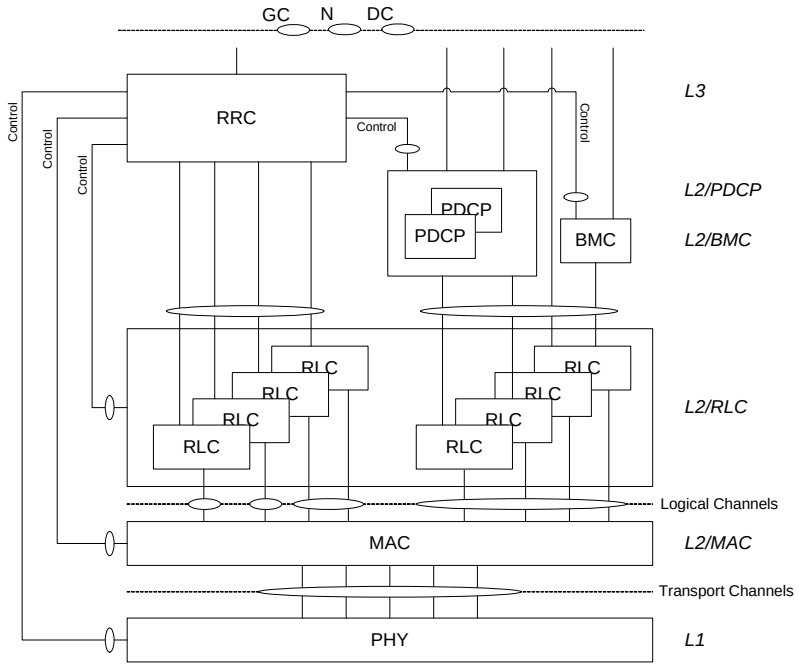


Figure 18: Architecture of the radio interface.

UMTS is the European branch of the family of the third generation wireless communications technologies IMT-2000 [3GPP]. The protocol architecture at the radio interface is shown in Figure 18 [3GPP 25.301]. In UMTS there are protocol conformance Test Suites for RLC, MAC, ICCPR, BMC, RRC, SMS (Short Message Service) and NAS [3GPP 34.123-3]. This section describes those aspects that are an improvement with respect to the design of the Bluetooth Test Systems described in the previous chapter.

1.10.1 Conformance Test System

The structure of the Protocols Module is shown in Figure 19 [COLA02]. It contains three blocks, two of which model the RLC and MAC Layers, while the third contains the elements that are optional in the Test Methods and allows selecting various configurations for internal testing. It is possible to handle up to eight cells within the same SDL system. Globally, the behavior of the Protocols Module is governed by the Physical Layer clock. The trace generation mechanism has been improved [ICUM94].

The RLC Layer [3GPP 25.322] has been modeled as a block type (Figure 20) where for each radio bearer a process is dynamically created to control its operation [COBA02]. It consists of a control process, two multiplexers at the top and bottom frontiers, a process type for each type of transfer service and a process for broadcast data. The buffers at RLC Layer are handled through an external library written in C.

The MAC Layer [3GPP 25.321] has been modeled as a process type whose behavior is described by services ([SORE03b]). To handle multiple cells each variable is declared as an array with as many positions as cells.

The decision on the design philosophy used in the RLC and MAC Layers has been taken after analyzing the benefits obtained by the use of processes or services [SORE03b]. Two aspects related with the speed of two alternatives were studied: signal communications and exchange of variables. Measurements ([MORI04], [ICUM17], [ICUM25]) have been conducted with a SDL model where a process communicates with

SPICUM servicio de publicaciones

64 instances of another process and the equivalent model using services. The platform used was a 1 GHz Pentium III with Windows 2000 operating system.

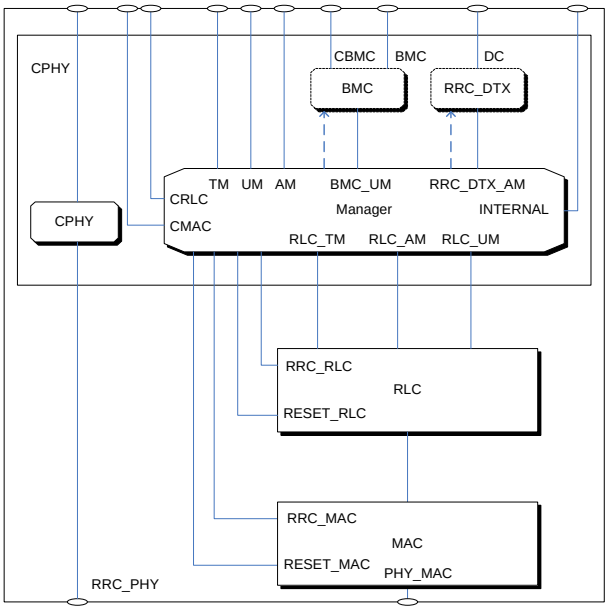


Figure 19: Structure of the Protocols Module in UMTS Test Systems.

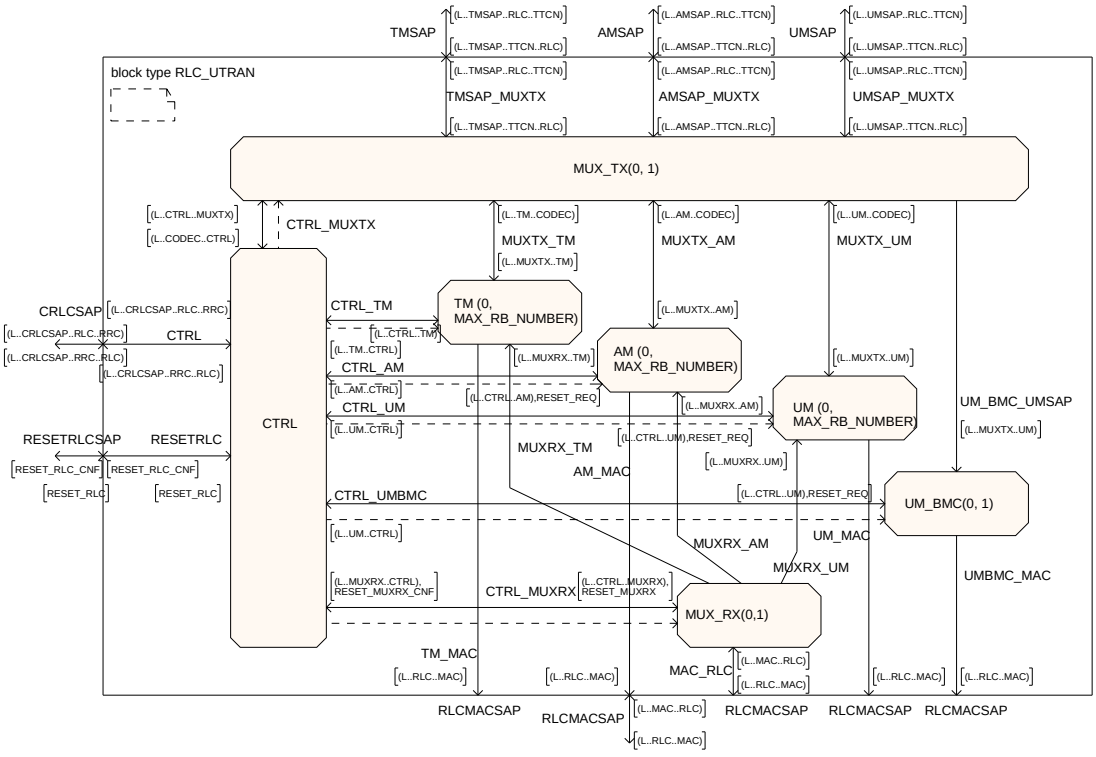


Figure 20: Structure of the RLC Layer.

SPICUM servicio de publicaciones

The external interfaces of layers present in the Protocols Module are specified in standards [3GPP 25.322] (RLC), [3GPP 25.321] (MAC) and [3GPP 25.302] (PHY), except for the control interface of the MAC Layer (CMAC), which is obtained from the Test Suites. Test Suites have been designed so that the end-to-end codification of RRC, BMC and NAS PDUs must be performed in the Lower Subsystem, adding at the same integrity protection. Our control interface allows selecting different configurations in the Protocols Module.

Communication between the Protocols Module and Physical Layer Module is realized through the Protocol Adapter Module, unlike how it was done in the Test Systems for DECT and Bluetooth. The Input/Output Management component has been modified to handle this interface. The coding of the primitives uses a format aligned to 32 bits [ICUM41]. Each field of the primitive is coded with a LV (Length-Value) scheme.

The components of the Test System run on different platforms. The Operation and Administration Subsystem and the Test Subsystem run on a Windows platform. The Protocols Module runs on a Linux operating system with kernel 2.4.20-18.9 [KERN03].

1.10.1.1 Tests

The module tests have been performed on the SDL system shown in Figure 21. The group formed by the Protocols Module and the Physical Layer emulators has been called Virtual Test System. This system allows testing the behavior of Layer 2 and above protocols without the need for a real Physical Layer Module.

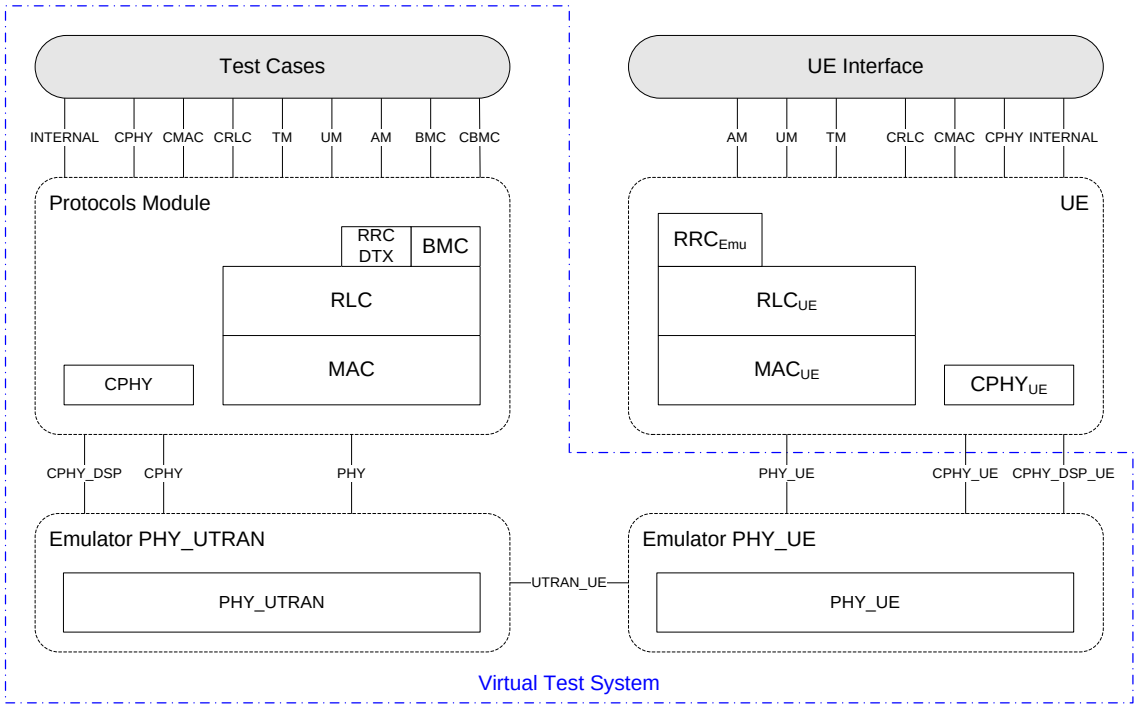


Figure 21: Architecture of the system used for the module tests.

The performance of the Protocols Module has been assessed on the Linux 2.4.20 (Pentium III 533 MHz) and Windows 2000 (Pentium IV 1 GHz) operating systems [SORE03b]. The transfer speeds considered have been the ones used by the bearers

SPICUM servicio de publicaciones

configured by the standard Test Cases: 12.2 kbps, 38.4 kbps and 384 kbps. Figure 22 shows the results for a speed of 384 kbps.

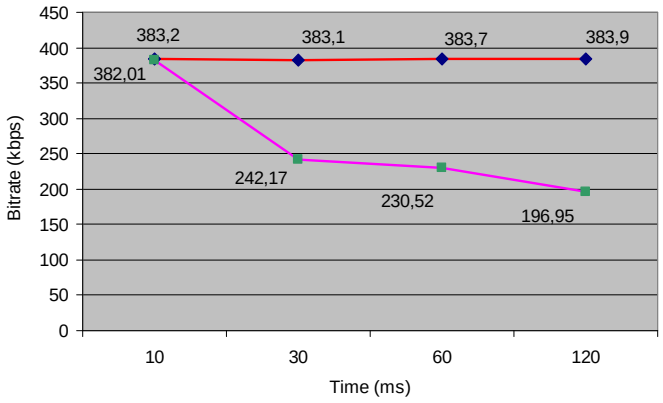


Figure 22: Transfer speeds achieved with a bearer configuration of 384 kbps in Windows and Linux.

1.10.2 Interoperability Test System

Following the line initiated in Bluetooth, an interoperability Test System for the UMTS RRC Layer has been built. The design process has been partially formalized using the development environment and integrating tools for automatic code generation from the conformance Test Suites. The Test System architecture is shown in Figure 23. The functionality of the layers below RRC is provided by the Lower Subsystem described in the previous section.

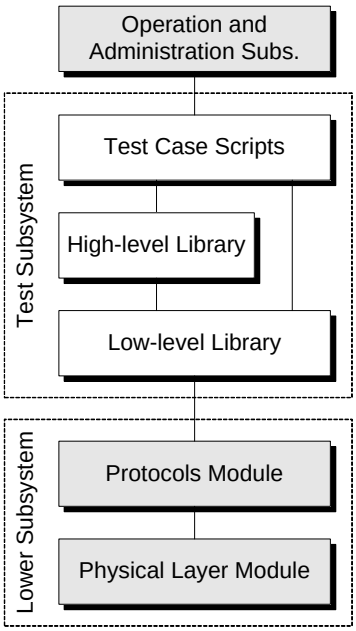


Figure 23: Architecture of the interoperability Test System.

SPICUM servicio de publicaciones

The Test System can be customized by the user, who can define and build its own specific test scenarios. The user can use two libraries, implemented in C, depending on the flexibility required. The high level interface provides functions equivalent to the TTCN Test Steps, which are used to model Test Cases. For example, the user can initiate the transmission of broadcast information or make a call.

The low level interface provides basic services, such as the transmission and reception of primitives handling of data types, encryption, timer management, assignment of verdicts, and even direct access to services in the top frontier of the Lower Subsystem. As much as possible is reused from the code generated by the development environment and the existing tools, incorporating both the TTCN Engine of the Test Subsystem and the Test Adapter and Test Management Modules, as well as part of the code generated from the Test Suites. The handling of data types has been encapsulated to offer a more convenient interface. These functions are generated automatically from the conformance Test Suites.

11 Radio Test Systems

This chapter demonstrated the application of Design Methodology to the field of radio Test Systems. As application examples radio Test Systems for Bluetooth and UMTS have been built ([PONC07a], [PONC07b], [GOME01a]) following the previously presented architecture. Both the use of the primitives of the interface proposed for the control of the instrumentation as the implemented Instrumentation Access Module have been validated.

The Test Parameters were obtained from the Test Specifications of each technology ([RF BTEST], [3GPP 34.121]). To implement algorithms difficult to model in TTCN, such as those for signal processing, TSO functions have been defined. The set of these functions could be generalized to define a library of functions to be used in the modeling of radio Test Suites.

1.11.1 Radio Test System for Bluetooth

The radio Test Cases for Bluetooth [BTEST RF] determine the EUT compliance in areas such as transmitted power and its spectral density, the modulation used (2 GFSK), off-band emissions and reception. Measurement procedures are described in [EN 300 328].

Thirteen radio Test Cases have been implemented ([GOME01b], [LOBA02]) from the transmission (TRM) and reception (RCV) groups. The instrumentation used for the transmission tests has been a spectrum analyzer, model FSIQ26, and for the reception tests a signal generator, model SMIQ03B, and an I/Q modulator, model AMIQ. The EUT must be activated in the transmission mode except for the reception tests, in which it must be activated in loopback mode.

1.11.2 Radio Test System for UMTS

The UMTS radio Tests Cases [3GPP 34.121] verify the EUT compliance for the transmission, reception, performance and radio resource management aspects. The measurement procedures and configurations are described in [3GPP 34.108]. Fourteen transmission Tests Cases have been implemented for the FDD mode [VALE02].

The instrumentation used has been a spectrum analyzer, model FSIQ26, complemented with a continuous wave generator, model ESG-D2000, to generate interferences. Test

SPICUM servicio de publicaciones

Cases are performed on a DPCH (Dedicated Physical Channel) uplink reference channel. Calls are established according to the generic setup procedure indicated in Section E.3.1 in [3GPP 34.121].

As an example of this modeling, the Thesis describes in detail the implementation of Test Case TRM/09, which verifies that the EUT complies with the limits of the established spectral emission mask [PONC07a].

12 Conclusions and Future Research

This Thesis describes a Design Methodology for Test Systems which can be used for both protocol and physical layer tests. The design process is based in a set of basic principles: use of ITU languages and notations, independence from the platform execution and use of commercial tools.

An efficient design of Test Systems requires a proper scheduling and structuring of its activities to cope with there complexity. To lower such complexity, we have defined a generic architecture, identifying its components and associated responsibilities, and a set of tools, which simplify the development being an integral part of the development process or easing the generation of some component. The most relevant results are mentioned next.

The Design Methodology structures the development process in eight stages, from its definition up to the attainment of an executable Test System. Each stage encompasses one or more activities; for each of them, we have indicated the input models and documents from which the output models and documents are generated. The dependencies between activities have been described as a lineal advance, though the design process is essentially iterative. Taking this methodology as a base, it is possible to make a more precise scheduling of the development process.

The Methodology assumes a design based in the ITU family of languages, such as SDL, TTCN and ASN.1. These languages allow doing a high-level abstract design. The adaptation of the abstract models to a specific execution platform is achieved via components external to them. The Methodology does not depend on specific commercial tools, and so it has been shown with the use of different development environments. However, its practical use must take into account the characteristics of the available commercial frameworks, which has influenced some of the design decisions.

The less detailed stage is that in which the Protocols Module is designed, where only some very high-level hints have been provided. The realization of these modules is a software design process, where several development methodologies already exist. In this Thesis methodology SOMT has been evaluated; it suggests a set of activities for the modeling with SDL of software systems. It has been studied how these activities are influenced when systems based on standards are designed. It is shown that some of the proposed activities are no longer needed, as they have been carried out during the standard development process. As a result, a new methodology, M-SOMT, is proposed, which simplifies the set of required activities and slightly modifies the target of some of the others. M-SOMT has been applied to the modeling of the DECT Network Layer.

The defined architecture consists of generic components, reusable without modification in different Test Systems, and specific components, which must be implemented each time. This provides a development and operation framework that can be used for all

SPICUM servicio de publicaciones

kinds of tests. The control and management of the Test Cases has been separated from their behavior and, at the same time, the modules that realize this behavior have been separated from the underlying protocols. This modularity allows multiple configurations in the deployment and the selection of the most appropriate execution platform for each component.

The architecture is independent from the execution platform and internally has been structured so that all functionality which depends from these platforms is grouped in the same module. Adapting this module to the new platform, it is possible to port the Test System without modifying the other elements. As examples, the Linux, Windows, Unix and DSP/BIOS platforms have been used.

An efficient design requires the use of tools that help producing a high-level design and that automate part of the tasks. Where the commercial tools do not offer a proper support, we have developed our own tools. These tools can be classified in two categories: reusable components, which are part of the architecture, and automatic generators, which generate interfaces between components of the architecture. These tools are adapted to the characteristics of the commercial environments, thus providing an integrated development environment

To validate the Design Methodology, protocol conformance Test Systems for DECT, Bluetooth and UMTS have been built. For the last two technologies, the result has been a commercial Test System. For Bluetooth, some of the developments have been made under an alternative commercial tool. The ideas, methods and tools used for conformance testing are also applicable to interoperability Test Systems. This has been demonstrated with the development of Test Systems for Bluetooth and UMTS. The designed Test Systems have been ported to several different platforms.

Although the Conformance Testing Methodology does not cover the area of physical layer tests, the test specification in both fields is performed follows the same guidelines. Thus, the Design Methodology has been applied to the development of physical layer Test Systems. This allows providing the same operation environment independently of the kind of test. It has been show that the physical layer tests can be modeled using the TTCN notation, the standard for the specification of protocol tests, and that this model can be made independent from specific measurement instrumentation. To achieve this, a generic interface has been defined between the physical layer tests and the instrumentation. This could be the way for physical layer tests to reach the same degree of formalization as that provided by protocol tests. Conformance physical layer Test Systems have been built for Bluetooth and UMTS.

1.12.1 Future Research

The work described in this Thesis can be continued via several paths, attempting to improve the design process efficiency and incorporating the probable future characteristics that Test Systems will have to provide.

On modeling languages. The Design Methodology can be adapted to the latest languages and notations that have appeared both in the testing field and in systems design in general. I am specifically talking about notations TTCN-3 and UML. In general, to use TTCN-3 in the design process, the interfaces of each component of the architecture suffer no semantic modification, although they need syntactic changes, but it is necessary to adapt the tools to the characteristics of the code generated by the development environments.

SPICUM servicio de publicaciones

On the other hand, it would be interesting to adapt the models and documents used along the Design Methodology to the UML notation, which is already supported by development environments with enough functionality. In this point, a higher integration between SDL and UML, as ITU is attempting, will be very positive, using UML for definition and documentation tasks and using SDL for the behavior description. Development environments which expedite a design process based on both notations in a truly integrated way are still expected.

On execution platforms. In this work it has been assumed that the platform is powerful enough, and thus the design has been carried out without tight constraining performance requirements. However, thinking on the portability of these systems, in the sense of geographic mobility, this aspect turns to be truly relevant. Thus, an improvement on the design process would be to tackle possible reductions in the size of code generated by the development environments, to decrease the required memory, and its speed optimization, to use less powerful processors and thus decrease power requirements. A small prototype has been made porting the DECT Test Systems to a DSP/BIOS platform.

On physical layer tests. In the area of physical layer tests the emphasis should be put on reaching a formalization level of the Test Specifications such as that provided for protocol testing. This demands new modeling paradigms that can be accepted both by suppliers and laboratories. It will be necessary to define an adequate test interface, a proposal of it has been shown in this Thesis, and to agree mechanisms for dynamic configuration of the measurement instrumentation.

On distributed tests. System testing is moving towards a distributed approach, where a central component assigns the final verdict, but the interaction with the Implementation Under Test is carried out through manifold test components located in different nodes of the network. Adapting the architecture used in this Thesis is quite straightforward, as the interaction between test components is handled by the code generated by the development environment. The Protocols Module must also be distributed and it would require creating several instances, which can be of the same or different type.

SPICUM servicio de publicaciones

BIBLIOGRAFÍA

[3GPP 23.002] 3GPP, TS 23.002, *Digital cellular telecommunications system (Phase 2+); Universal Mobile Telecommunications System (UMTS); Network Architecture*, v3.6.0, 2002.

[3GPP 23.101] 3GPP, TS 23.101, *3rd Generation Partnership Project; Technical Specification Group Services and System Aspects General UMTS Architecture*, v3.1.0, 2000.

[3GPP 25.201] 3GPP, TS 25.201, *3rd Generation Partnership Project; Technical Specification Group Radio Access Network; Physical layer - General description*, v3.4.0, 2002.

[3GPP 25.301] 3GPP, TS 25.301, *3rd Generation Partnership Project; Technical Specification Group Radio Access Network; Radio Interface Protocol Architecture*, v3.11.0, 2002.

[3GPP 25.302] 3GPP, TS 25.302, *3rd Generation Partnership Project; Technical Specification Group Radio Access Network; Services provided by the physical layer*, v3.16.0, 2003.

[3GPP 25.321] 3GPP, TS 25.321, *3rd Generation Partnership Project; Technical Specification Group Radio Access Network; Medium Access Control (MAC) protocol specification*, v3.16.0, 2003.

[3GPP 25.322] 3GPP, TS 25.322, *3rd Generation Partnership Project; Technical Specification Group Radio Access Network; Radio Link Control (RLC) protocol specification*, v3.16.0, 2003.

[3GPP 25.323] 3GPP, TS 25.323, *3rd Generation Partnership Project; Technical Specification Group Radio Access Network; Packet Data Convergence Protocol (PDCP) Specification*, v3.10.0, 2002.

[3GPP 25.324] 3GPP, TS 25.324, *3rd Generation Partnership Project; Technical Specification Group Radio Access Network; Broadcast/Multicast Control (BMC)*, v3.7.0, 2003.

[3GPP 25.331] 3GPP, TS 25.331, *3rd Generation Partnership Project; Technical Specification Group Radio Access Network; Radio Resource Control (RRC) protocol specification*, v3.16.0, 2003.

[3GPP 34.108] 3GPP, TS 34.108, *3rd Generation Partnership Project; Technical Specification Group Terminals; Common test environments for User Equipment (UE) conformance testing*, v3.16.0, 2004.

[3GPP 34.121] 3GPP, TS 34.121, *3rd Generation Partnership Project; Technical Specification Group Terminals; Terminal conformance specification; Radio transmission and reception (FDD)*, v3.14.0, 2003.

[3GPP 34.123] 3GPP, TS 34.123, *3rd Generation Partnership Project; Technical Specification Group Terminals; User Equipment (UE) conformance specification*, v3.5.0/v3.6.0, 2004.

SPICUM servicio de publicaciones

[3GPP 34.123-1] 3GPP, TS 34.123-1, *3rd Generation Partnership Project; Technical Specification Group Terminals; User Equipment (UE) conformance specification; Part 1: Protocol conformance specification*, v3.5.0, 2001.

[3GPP 34.123-2] 3GPP, TS 34.123-2, *3rd Generation Partnership Project; Technical Specification Group Terminal; User Equipment (UE) conformance specification; Part 2: Implementation Conformance Statement (ICS) proforma specification*, v3.5.0, 2001.

[3GPP 34.123-3] 3GPP, TS 34.123-3, *3rd Generation Partnership Project; Technical Specification Group Terminals; User Equipment (UE) conformance specification; Part 3: Abstract Test Suite (ATS)*, v3.6.0, 2004.

[3GPP 35.201] 3GPP, TS 35.201, *3rd Generation Partnership Project; Technical Specification Group Services and System Aspects; 3G Security; Specification of the 3GPP Confidentiality and Integrity Algorithms; Document 1: f8 and f9 Specification*, v3.2.0, 2001.

[3GPP 35.202] 3GPP, TS 35.202, *3rd Generation Partnership Project; Technical Specification Group Services and System Aspects; 3G Security; Specification of the 3GPP Confidentiality and Integrity Algorithms; Document 2: KASUMI Specification*, v3.1.2, 2001.

[3GPP 35.203] 3GPP, TS 25.203, *3rd Generation Partnership Project; Technical Specification Group Services and System Aspects; 3G Security; Specification of the 3GPP Confidentiality and Integrity Algorithms; Document 3: Implementors' Test Data*, v3.1.2, 2001.

[3GPP] *3rd Generation Partnership Project*, http://www.3gpp.org/specs/specs.htm, 2008.

[ACACIA] *Acacia*, http://www.acacia-net.com/, 2004.

[ADAM31] Adamiecki, Karol, *Harmonograf*, Przegląd Organizacji, 1931.

[AHO86] A. V. Aho, R. Sethi, J. D. Ullman, *Compilers, Principles, Techniques, and Tools*, Bell Telephone Laboratories, 1986.

[ALAR01] L. Alarcón, *Modelado del Control de la Llamada en UMTS*, ETSI Telecomunicación, Universidad de Málaga, 2001.

[ALBA99] J. P. Albaladejo, J. Poncela, *Aplicación del SDL'92 al Diseño de un Sistema DECT*, XIV Simposium Nacional de la Unión Científica Internacional de Radio (URSI'99), Santiago de Compostela, 8-10 Septiembre 1999.

[ALBA00a] J. P. Albaladejo, J. Poncela, J. T. Entrambasaguas, *Adaptation of SOMT to the Development of Systems Based Upon a Standard*, 2nd Workshop of the SDL Forum Society on SDL and MSC (SAM2000), Grenoble (Francia), 26-28 Junio 2000.

[ALBA00b] J. P. Albaladejo, *Aplicación de SDL'92 al Diseño de Sistemas de Comunicaciones*, ETSI Telecomunicación, Universidad de Málaga, 2000.

SPICUM servicio de publicaciones

[ALGA94] E. Algaba, C. F. Cano, J. I. Sanz, O. Valcárcel, *HARPO: Herramientas para la Realización de Pruebas OSI*, Comunicaciones de Telefónica I+D, vol. 5, nº 1, 1994.

[ALON97] A. Alonistioti, P. Kostarakis, *Integration and SDL-Based Modelling of Generis UMTS Handover Service Architecture and IN*, Int. Journal of Communication Systems, vol. 10, nº 3, John Wiley & Sons, pp. 139-146, 1997.

[ALTERA] *Altera Corporation*, http://www.altera.com, 2006.

[AMIQ] Rohde & Schwarz, *I/Q Modulation generator AMIQ, Service manual*, 1110.3339, 2000.

[AMYO99] D. Amyot, R. Andrade, L. Logrippo, J. Sincennes, Z. Yi, *Formal Methods for Mobility Standards*, IEEE Emerging Technology Symposium on Wireless CommunicationsSystems, pp. 14.1-14.7, 1999.

[ANITUM] Anite, *Wireless handsets conformance testing*, http://www.anite.com/wireless-handset-conformance-testing-anite.html?Itemid=58, 2007.

[ANRIUM] Anritsu, *Conformance Test Systems*, http://www.eu.anritsu.com/products/default.php?c=5#67, 2007.

[ANTL95] T. J. Parr, R. W. Brown, *ANTLR: a Predicated-LL(k) Parser Generator*, Software Practice & Experience, vol. 25, pp. 789-810, 1995.

[ANTL99] T. J. Parr, *ANother Tool for Language Recognition*, http://www.antlr.org/, 1999.

[ARMI97] A. P.-G. Eberlein, F. Halsall, *Telecommunications Service Development: A Design Methodology and Its Intelligent Support*, Journal of Engineering Applications of Artificial Intelligence, vol. 10, nº 6, pp. 647-663, 1997.

[AT4W] *AT4 wireless*, http://www.at4wireless.com/, 2007.

[ATCOM] *Hayes command set*, http://en.wikipedia.org/wiki/AT_command_set, 2006.

[AXIS01] *AXIS Open BT Stack*, http://sourceforge.net/projects/openbt/, 2001.

[BADR07] I. Badr, *Agile Modeling: Rapid Enterprise IT Prototyping and Development*, Telelogic, v1.0, Septiembre 2007.

[BAÑO02] J. Baños, *Testing of Bluetooth Products in the Industrial Environment*, 28th Annual Conference of the IEEE Industrial Electronics Society (IECON), Sevilla (España), 5-8 Noviembre 2002.

[BÄR92] U. Bär, J. M. Schneider, *Automated Validation of TTCN Test Suites*, Protocol Specification, Testing and Verification XII, IFIP, pp. 279-295, Elsevier Science, 1992.

[BEIZ90] B. Beizer, *Software Testing Techniques*, 2ª ed., Van Nostrand Reinhold, ISBN 0-442-20672-0, 1990.

SPICUM servicio de publicaciones

[BERT90] H. V. Bertine, W. B. Elsner, P. K. Verma, *Overview of Protocol Testing Programs, Methodologies, and Standards*, vol. 69, nº 1, AT&T Technical Journal, 1990.

[BERW96] P. Berwing, *DECT Poised for the Public Network*, Telcom Report International, vol. 19, nº 3, 1996.

[BHAM98] A. Bhamidipaty, T. A. Proebsting, *Very Fast YACC-Compatible Parsers (For Very Little Effort)*, Software – Practice and Experience, vol. 28, nº 2, John Wiley & Sons, pp. 181-190, 1998.

[BITE] *Bluetooth Qualification Tester (BITE)*, AT4 wireless, http://www.at4wireless.com/web_esp/sistemas/soluciones_ensayo/bite, 2003.

[BLAN02] I. Blanco, *Modelado de la Gestión de la Movilidad en UMTS*, ETSI Telecomunicación, Universidad de Málaga, 2002.

[BOOC94] G. Booch, *Object-Oriented Analysis and Design with Applications*, Addison-Wesley, ISBN 0-8053-5340-2, 1994.

[BOOC95] G. Booch, J. Rumbaugh, *Unified Method for Object-oriented Development. Documentation Set*, v0.8, Rational Software Corporation, 1995.

[BORN98] M. Born, A. Hoffman, *An Object-Oriented Design Methodology for Distributed Services*, Proceedings of Technology of Object-Oriented Languages and Systems (TOOLS 28), pp. 52-64, 1998.

[BOUR01] C. Bourhfir, E. Aboulhamid, R. Dssouli, N. Rico, *A test case generation approach for conformance testing of SDL systems*, Computer Communications, vol. 24, nº 3-4, pp. 319-333, 2001.

[BRÆK93] R. Bræk, Ø. Haugen, *Engineering Real Time Systems*, Prentice-Hall, ISBN 0 -13-034448-6, 1993.

[BRÆK99] R. Bræk, J. Gorman, Ø. Haugen, G. Melby, B. Møller-Pedersen, R. Sanders, *TIMe – The Integrated Method*, SINTEF, Noruega, 1999.

[BREU95] P. Breuer, J. Bowen, *A PREttier Compiler-Compiler: Generating High Order Parsers in C*, Software Practice and Experience, pp. 1267-1297, Noviembre 1995.

[BREU97] P. Breuer, *PRECC – A PREttier Compiler-Compiler*, http://vl.fmnet.info/precc/, 1997.

[BRIN92] E. Brinksma, *What is the Method in Formal Methods?*, Formal Description Techniques IV, IFIP, pp. 33-50, Elsevier Science, 1992.

[BT BB] Bluetooth SIG, *Specification of the Bluetooth System – Core – Part B – Baseband Specification*, vol.1, v1.1, Febrero 2001

[BT CORE] Bluetooth SIG, *Specification of the Bluetooth System – Core*, vol.1, v1.1, Febrero 2001.

[BT GAP] Bluetooth SIG, *Specification of the Bluetooth System – Profiles – Part K:1 – Generis Access Profile*, vol.1, v1.1, Febrero 2001.

SPICUM servicio de publicaciones

[BT GOEP] Bluetooth SIG, *Specification of the Bluetooth System – Profiles – Part K:10 – Generis Object Exchange Profile*, vol.1, v1.1, Febrero 2001.

[BT HCI] Bluetooth SIG, *Specification of the Bluetooth System – Core – Part H – Host Controller Interface Functional Specification*, vol.1, v1.1, Febrero 2001.

[BT HEAD] Bluetooth SIG, *Specification of the Bluetooth System – Profiles – Part K:6 – Headset Profile*, vol.1, v1.1, Febrero 2001.

[BT HID] Bluetooth SIG, *Human Interface Device (HID) Profile*, 2003.

[BT L2CAP] Bluetooth SIG, *Specification of the Bluetooth System – Core – Part D – Logical Link Control and Adaptation Protocol Specification*, vol.1, v1.1, Febrero 2001.

[BT LANAP] Bluetooth SIG, *Specification of the Bluetooth System – Profiles – Part K:9 – LAN Access Profile*, vol.1, v1.1, Febrero 2001.

[BT LM] Bluetooth SIG, *Specification of the Bluetooth System – Core – Part C – Link Manager Protocol*, vol.1, v1.1, Febrero 2001.

[BT OPP] Bluetooth SIG, *Specification of the Bluetooth System – Profiles – Part K:11 – Object Push Profile*, vol.1, v1.1, Febrero 2001.

[BT OVER] Bluetooth SIG, *Comprehensive Description of the Bluetooth System*, 1998.

[BT PROF] Bluetooth SIG, *Specification of the Bluetooth System – Profiles*, vol.1, v1.1, Febrero 2001.

[BT RFCOMM] Bluetooth SIG, *Specification of the Bluetooth System – Core – Part F:1 – RFCOMM with TS 07.10*, vol.1, v1.1, Febrero 2001.

[BT SDP] Bluetooth SIG, *Specification of the Bluetooth System – Core – Part E – Service Discovery Protocol*, vol.1, v1.1, Febrero 2001.

[BT SPP] Bluetooth SIG, *Specification of the Bluetooth System – Profiles – Part K:5 – Serial Port Profile*, vol.1, v1.1, Febrero 2001.

[BT TCS] Bluetooth SIG, *Specification of the Bluetooth System – Core – Part F:3 – Telephony Control Protocol Specification*, vol.1, v1.1, Febrero 2001.

[BTEST BB] Bluetooth SIG, Bluetooth Test & Interoperability Testing Group, *Test Specification: Part B. Test Suite (TSS) and Test Purposes (TP) for Baseband*, v0.9, Mayo 2002.

[BTEST GAP] Bluetooth SIG, Bluetooth Test & Interoperability Testing Group, *Test Specification: Part K:1. Test Suite (TSS) and Test Purposes (TP) for Generic Access Profile*, v0.9, Mayo 2002.

[BTEST HEAD] Bluetooth SIG, Bluetooth Test & Interoperability Testing Group, *Profile Test Specification: Part K:6. Test Suite (TSS) and Test Purposes (TP) for Headset Profile*, v1.1, Julio 2002.

SPICUM servicio de publicaciones

[BTEST L2CAP] Bluetooth SIG, Bluetooth Test & Interoperability Testing Group, *Test Specification: Part D. Test Suite (TSS) and Test Purposes (TP) for Logical Link and Adaptation Protocol*, v0.9, Mayo 2002.

[BTEST LM] Bluetooth SIG, Bluetooth Test & Interoperability Testing Group, *Test Specification: Part C. Test Suite (TSS) and Test Purposes (TP) for Link Manager*, v0.9, Mayo 2002.

[BTEST PRO] Bluetooth SIG, Bluetooth Test & Interoperability Testing Group, *Profile Test Specification: Part K. Test Suite (TSS) and Test Purposes (TP)*, v1.1, Julio 2002.

[BTEST RF] Bluetooth SIG, *Test Specification RF*, v1.1, 2001.

[BTEST SDP] Bluetooth SIG, Bluetooth Test & Interoperability Testing Group, *Test Specification: Part E. Test Suite (TSS) and Test Purposes (TP) for Service Discovery Protocol*, v0.9, Mayo 2002.

[BTEST SPP] Bluetooth SIG, Bluetooth Test & Interoperability Testing Group, *Test Specification: Part K:5. Test Suite (TSS) and Test Purposes (TP) for Serial Port Profile*, v0.9, Mayo 2002.

[BURK96] R. J. Hathaway III, *What Are The Current Object-Oriented Methodologies?*, Object Magazine Online, Cyberdyne Object Systems, http://burks.brighton.ac.uk/burks/pcinfo/progdocs/oofaq/s37.htm, 1997.

[BUSC96] F. Buschmann, R. Meunier, H. Rohnert, P. Sommerlad, M. Stal, *Pattern Oriented Software Architecture – A System of Pattern*, John Wiley and Sons Ltd, ISBN 0-471-95869-7, 1996.

[BUSH90] M. Bush, K. Rasmussen, F. Wong, *Conformance Testing Methodologies for OSI Protocols*, AT&T Technical Journal, vol. 69, nº 1, pp. 84-100, 1990.

[CABE00] F. Cabello, *Herramientas Gráficas para el Lenguaje ODL*, ETSI Telecomunicación, Universidad de Málaga, 2000.

[CARD05] C. Cárdenas, *Software de Control para Dispositivos Bluetooth Multifabricante de Interfaz USB*, ETSI Telecomunicación, Universidad de Málaga, 2005.

[CATB07] *The Jargon File*, http://www.catb.org/jargon/html/T/tool.html, 2007.

[CAVA94] A. R. Cavalli, J. P. Favreau, M. Phalippou, *Formal Methods in Conformance Testing: Results and Perspectives*, Protocol Test Systems VI, IFIP, pp. 3-17, Elsevier Science, 1994.

[CESP99] A. M. Céspedes, *Implementación del Nivel de Enlace de un Sistema DECT*, ETSI Telecomunicación, Universidad de Málaga, 1999.

[CHAN93] S. T. Chanson, Q. Li, *On Inconclusive Verdict in Conformance Testing,* Protocol Test Systems V, IFIP, pp. 81-92, Elsevier Science, 1993.

[CINDER] *Cinderella ApS*, http://www.cinderella.dk/, 2004.

SPICUM servicio de publicaciones

[COAD91] P. Coad, E. Yourdon, *Object-Oriented Analysis*, 2ª ed., Prentice Hall, ISBN 0-13-629981-4, 1990.

[COBA02] M. A. Cobalea, *Modelado del Control del Enlace en UMTS*, ETSI Telecomunicación, Universidad de Málaga, 2002.

[COLA00] J. F. Colás, *Analizador Sintáctico y Semántico de Sistemas SDL*, ETSI Telecomunicación, Universidad de Málaga, 2000.

[COLA02] J. F. Colás, J. M. Pérez, J. Poncela, J. T. Entrambasaguas, *Implementation of UMTS Protocol Layers for the Radio Access Interface*, 3rd Workshop of the SDL Forum Society on SDL and MSC (SAM2002), Aberystwyth (UK), 24-26/Junio/2002.

[COLA03] J. F. Colás, J. M. Pérez, J. Poncela, J. T. Entrambasaguas, *Implementation of UMTS Protocol Layers for the Radio Access Interface*, Telecommunications and Beyond: The Broader Applicability of SDL and MSC, Lecture Notes in Computer Science, Springer Verlag, ISBN 3-540-00877-2, pp. 74-89, 2003.

[COLE94] D. Coleman, et. al., *Object-Oriented Development - The Fusion Method*, Prentice-Hall, ISBN 0-13-338823-9, 1994.

[CONT97] Contrato I+D, *Herramienta para la automatización de sistemas de medida de telecomunicaciones*, J. T. Entrambasaguas, Centro de Tecnología de las Comunicaciones S.A., Grupo de Ingeniería de Comunicaciones, Octubre 1997 – Abril 1998.

[CONT98] Contrato I+D, *Herramienta para pruebas de sistemas y redes de telecomunicación*, J. T. Entrambasaguas, Centro de Tecnología de las Comunicaciones S.A., Grupo de Ingeniería de Comunicaciones, Junio 1998 – Mayo 1999.

[CONT99a] Contrato I+D, *Sistemas de prueba para comunicaciones móviles globales por satélite*, J. T. Entrambasaguas, Centro de Tecnología de las Comunicaciones S.A., Grupo de Ingeniería de Comunicaciones, Marzo 1999 – Agosto 1999.

[CONT99b] Contrato I+D, *Herramienta para pruebas de sistemas y redes de telecomunicación, Ampliación 1*, J. T. Entrambasaguas, Centro de Tecnología de las Comunicaciones S.A., Grupo de Ingeniería de Comunicaciones, Junio 1999 – Diciembre 2001.

[CONT00] Contrato I+D, *Centro de competencia de sistemas de comunicaciones móviles en el Centro de Investigación de la Universidad de Málaga en el Parque Tecnológico de Andalucía en Málaga (fases 2-7)*, C. Camacho, Plan nacional de I+D. Proyectos cofinanciados con fondos FEDER, Programa Nacional de Tecnologías de la Información y de las Telecomunicaciones, Colaboración Nokia – Ingeniería de Comunicaciones, Julio 2000 – Julio 2003.

[CONT02] Contrato I+D, *Herramienta para pruebas de sistemas y redes de telecomunicación, Ampliación 2*, J. T. Entrambasaguas, Centro de Tecnología de las Comunicaciones S.A., Grupo de Ingeniería de Comunicaciones, Enero 2002 – Diciembre 2002.

SPICUM servicio de publicaciones

[CONT03a] Contrato I+D, *Tecnología base de comunicaciones*, J.T. Entrambasaguas, Centro de Tecnología de las Comunicaciones S.A., Grupo de Ingeniería de Comunicaciones, Enero 2003 – Junio 2004.

[CONT03b] A. Contreras, *Modelado en SDL de la Capa MAC de UMTS*, ETSI Telecomunicación, Universidad de Málaga, 2003.

[COOP03] J. Cooperstein, *Linux Kernel 2.6: New Features – I*, Axian, 2003.

[COX95] D. Cox, *Wireless Personal Communications: What Is It?*, IEEE Personal Communications, vol. 2, nº 2, pp. 20-35, 1995

[CROW93] M. J. Crowther, *Modelling and Specifying C7 Using SDL*, BT Technology Journal, vol. 11, nº 4, pp. 9-15, 1993.

[CSR02] Cambridge Silicon Radio, *The Casira development system for Bluetooth*, http://www.csr.com/development/casira.htm, 2002.

[CYGWIN] *Cygwin*, http://www.cygwin.com/, 2001.

[DAHB90] A. Dahbura, K. Sabnani, M. Ü. Uyar, *Algorithmic Generation of Protocol Conformance Tests*, AT&T Technical Journal, vol. 69, nº 1, pp. 101-118, 1990.

[DANET] Danet GmbH, *TTCN Toolbox*, http://www.danet.com/, 2004

[DECT] ETSI, *ETSI DECT*, http://www.etsi.org/WebSite/Technologies/DECT.aspx, 2007.

[DELI01] ETSI, *ETSI Deliverables*, http://www.etsi.org/smp/documents/deliverables.htm, 2002.

[DELI02] ETSI, *ETSI Deliverables Presentation*, http://www.etsi.org/smp/presentations/Deliverables.ppt, 2002.

[DERR98] K. W. Derr, *Applying OMT*, Cambridge University Press, ISBN 0-13-231390-1, 1998.

[DEWE07] DECTWeb, *DECT Products*, http://www.dectweb.com/Products/products.htm, 2007.

[DIET02] F. Dietrich, J. P. Hubaux, *Formal Methods for Communication Services: Meeting the Industry Expectations*, Computer Networks, vol. 38, nº 1, pp. 99-120, 2002.

[DOMI99] M. J. Domínguez, *Emulador del Nivel MAC del Sistema DECT*, ETSI Telecomunicación, Universidad de Málaga, 1999.

[DRAE01] Real Academia Española, *Diccionario de la Real Academia Española*, http://www.rae.es/, 22ª ed., 2001.

[DSPC6201] Texas Instruments, *TMS320C6201 Digital Signal Processor*, http://focus.ti.com/docs/prod/folders/print/tms320c6201.html, 2004.

[DSSO99] R. Dssouli, K. Saleh, E. Aboulhamid, A. En-Nouaary, C. Bourhfir, *Test development for communication protocols: towards automation*, Computer Networks, vol. 31, nº 17, pp. 1835-1872, 1999.

SPICUM servicio de publicaciones

[EBDK01] Ericsson, *Ericsson Bluetooth Development Kit*, http://www.ericsson.com/se/microe/pdf/other/bluetooth/bluetooth_startkit.pdf, 2001.

[EG 202 307] ETSI, EG 202 237, *Methods for Testing and Specification (MTS);Internet Protocol Testing (IPT);Generic approach to interoperability testing*, v1.1.2, 2007.

[EK97] A. Ek, J. Grabowski, D. Hogrefe, R. Jerome, B. Koch, M. Schmitt, *Towards the Industrial Use of Validation Techniques and Automatic Test Generation Methods for SDL Specifications*, 8th International SDL Forum – Time for Testing, SDL, MSC and Trends (SDL '97), pp. 245-259, Evry (Francia), Septiembre 1997.

[EKAN95] A. Ek, *The SOMT Method*, Telelogic, Septiembre 1995.

[ELLS97] J. Ellsberger, D. Hogrefe, A. Sarma, *SDL: Formal Oriented Language for Communicating Systems*, Prentice-Hall, ISBN 0-13-621384-7, 1997.

[EN 300 328] ETSI, EN 300 328, *Electromagnetic compatibility and Radio spectrum Matters (ERM); Wideband Transmission systems; Data transmission equipment operating in the 2,4 GHz ISM band and using spread spectrum modulation techniques; Part 1: Technical characteristics and test conditions*, v1.3.1, 2001.

[EN 300 370] ETSI, EN 300 370, *Digital Enhanced Cordless Telecommunications (DECT); Global System for Mobile communications (GSM); DECT/GSM Interworking Profile (IWP); Access and mapping (protocol/procedure description for 3,1 kHz speech service)*, v1.3.1, 2001.

[EN 300 434] ETSI, EN 300 434, *Digital Enhanced Cordless Telecommunications (DECT); Integrated Services Digital Network (ISDN); DECT/ISDN interworking for end system configuration*, v1.2.1, 2001.

[EN 300 497] ETSI, EN 300 497, *Digital Enhanced Cordless Telecommunications (DECT); Common Interface (CI); Test Suite Structure (TSS) and Test Purposes (TP) and Test Case Library (TCL); Parts 1-9*, v0.3.2, 1999.

[EN 300 497-2] ETSI, EN 300 497-2, *Digital Enhanced Cordless Telecommunications (DECT); Common Interface (CI); Test Case Library (TCL); Part 2: Abstract Test Suite (ATS) for Medium Access Control (MAC) layer - Portable radio Termination (PT)*, v0.3.1, 1999.

[EN 300 497-3] ETSI, EN 300 497-3, *Digital Enhanced Cordless Telecommunications (DECT); Common Interface (CI); Test Case Library (TCL); Part 3: Abstract Test Suite (ATS) for Medium Access Control (MAC) layer - Fixed radio Termination (FT)*, v0.3.2, 1999.

SPICUM servicio de publicaciones

[EN 300 497-4] ETSI, EN 300 497-4, *Digital Enhanced Cordless Telecommunications (DECT); Common Interface (CI); Test Case Library (TCL); Part 4: Test Suite Structure (TSS) and Test Purposes (TP) - Data Link Control (DLC) layer*, v0.3.0, 1999.

[EN 300 497-5] ETSI, EN 300 497-5, *Digital Enhanced Cordless Telecommunications (DECT); Common Interface (CI); Test Case Library (TCL); Part 5: Abstract Test Suite (ATS) - Data Link Control (DLC) layer*, v0.3.0, 1999.

[EN 300 497-6] ETSI, EN 300 497-6, *Digital Enhanced Cordless Telecommunications (DECT); Common Interface (CI); Test Case Library (TCL); Part 6: Test Suite Structure (TSS) and Test Purposes (TP) - Network (NWK) layer - Portable radio Termination (PT)*, v0.3.2, 1999.

[EN 300 497-7] ETSI, EN 300 497-7, *Digital Enhanced Cordless Telecommunications (DECT); Common Interface (CI); Test Case Library (TCL); Part 7: Abstract Test Suite (ATS) for Network (NWK) layer - Portable radio Termination (PT)*, v0.3.0, 1999.

[EN 300 497-8] ETSI, EN 300 497-8, *Digital Enhanced Cordless Telecommunications (DECT); Common Interface (CI); Test Case Library (TCL); Part 8: Test Suite Structure (TSS) and Test Purposes (TP) - Network (NWK) layer - Fixed radio Termination (FT)*, v0.3.2, 1999.

[EN 300 497-9] ETSI, EN 300 497-9, *Digital Enhanced Cordless Telecommunications (DECT); Common Interface (CI); Test Case Library (TCL); Part 9: Abstract Test Suite (ATS) for Network (NWK) layer - Fixed radio Termination (FT)*, v0.3.2, 1999.

[EN 300 607-1] ETSI, EN 300 607-1, *Digital cellular telecommunications system (Phase 2+) (GSM); Mobile Station (MS) conformance specification; Part 1: Conformance specification (GSM 11.10-1 version 8.1.1 Release 1999)*, v8.1.1, 2000.

[EN 300 607-3] ETSI, EN 300 607-3, *Digital cellular telecommunications system (Phase 2+) (GSM); Mobile Station (MS) conformance specification; Part 3: Layer 3 Abstract Test Suite (ATS) (GSM 11.10-3 version 5.0.1 Release 1996)*, v5.0.1, 2000.

[EN 300 757] ETSI, EN 300 757, *Digital Enhanced Cordless Telecommunications (DECT); Low Rate Messaging Service (LRMS) including Short Messaging Service (SMS)*, v1.5.1, 2004.

[EN 300 757] ETSI, EN 300 757, *Digital Enhanced Cordless Telecommunications (DECT); Low Rate Messaging Service (LRMS) including Short Messaging Service (SMS)*, v1.5.1, 2004.

[EN 300 765] ETSI, EN 300 765, *Digital Enhanced Cordless Telecommunications (DECT); Radio in the Local Loop (RLL) Access Profile (RAP)*, v1.3.1, 2001.

SPICUM servicio de publicaciones

[EN 300 822] ETSI, EN 300 822, *Digital Enhanced Cordless Telecommunications (DECT); Integrated Services Digital Network (ISDN); DECT/ISDN interworking for intermediate system configuration; Interworking and profile specification*, v1.2.1, 2001.

[EN 300 824] ETSI, EN 300 824, *Digital Enhanced Cordless Telecommunications (DECT); Cordless Terminal Mobility (CTM); CTM Access Profile (CAP)*, v1.3.1, 2001.

[EN 301 238] ETSI, EN 301 238, *Digital Enhanced Cordless Telecommunications (DECT); Data Services Profile (DSP); Isochronous data bearer services with roaming mobility (service type D, mobility class 2)*, v1.3.1, 2001.

[EN 301 242] ETSI, EN 301 242, *Digital Enhanced Cordless Telecommunications (DECT); Global System for Mobile communications (GSM); DECT/GSM integration based on dual-mode terminals*, v1.2.2, 1999.

[EN 301 649] ETSI, EN 301 649, *Digital Enhanced Cordless Telecommunications (DECT);DECT Packet Radio Service (DPRS)*, v1.4.1, 2004.

[EN 301 650] ETSI, EN 301 650, *Digital Enhanced Cordless Telecommunications (DECT); DECT Multimedia Access Profile (DMAP); Application Specific Access Profile (ASAP)*, v1.2.1, 2002.

[ES 201 873] ETSI, ES 201 873, *Methods for Testing and Specification (MTS); The Testing and Test Control Notation version 3*, v1.1.1-v3.2.1, 2003-2007.

[ES 201 873-1] ETSI, ES 201 873-1, *Methods for Testing and Specification (MTS); The Testing and Test Control Notation version 3; Part 1: TTCN-3 Core Language*, v2.2.1, 2003.

[ES 201 873-5] ETSI, ES 201 873-5, *Methods for Testing and Specification (MTS); The Testing and Test Control Notation version 3; Part 5: TTCN-3 Runtime Interface (TRI)*, v1.1.1, 2003.

[ESG-D2000] Hewlett Packard, HP ESG-D2000A GB3704, *HP ESG-D Series Signal Generator User's Guide E4400-90081*, 2001.

[ESPI05] J. I. Espínola, *Aplicaciones Bluetooth para Acceso a Servicios Locales de Información Mediante Teléfonos Móviles*, ETSI Telecomunicación, Universidad de Málaga, 2005.

[ETG 029] EWOS, ETG 029, *Interoperability Vocabulary*, 1993.

[ETG 059] EWOS, ETG 059, OSE Profile Conformance Testing, 1995.

[ETR 056] ETSI, ETR 056, *Digital Enhanced Cordless Telecommunications (DECT); System description document*, v1, 1993.

[ETR 130] ETSI, ETR 130, *Methods for Testing and Specification (MTS); Interoperability and conformance testing A classification scheme*, v1, 1994.

SPICUM servicio de publicaciones

[ETR 141] ETSI, ETR 141, *Methods for Testing and Specification (MTS); Protocol and profile conformance testing specifications; The Tree and Tabular Combined Notation (TTCN) style guide*, v1, 1994.

[ETR 178] ETSI, ETR 178, *Digital Enhanced Cordless Telecommunications (DECT); A high level guide to the DECT standardization*, v2, 1997.

[ETR 183] ETSI, ETR 183, *Digital Enhanced Cordless Telecommunications (DECT); Conformance testing on DECT equipment*, v1, 1995.

[ETS 300 175] ETSI, ETS 300 175, *Digital Enhanced Cordless Telecommunications (DECT); Common Interface (CI)*, v2, 1996/97.

[ETS 300 175-1] ETSI, ETS 300 175-1, *Digital Enhanced Cordless Telecommunications (DECT); Common Interface (CI); Part 1: Overview*, v2, 1996.

[ETS 300 175-2] ETSI, ETS 300 175-2, *Digital Enhanced Cordless Telecommunications (DECT); Common Interface (CI); Part 2: Physical Layer (PHL)*, v2, 1996.

[ETS 300 175-3] ETSI, ETS 300 175-3, *Digital Enhanced Cordless Telecommunications (DECT); Common Interface (CI); Part 3: Medium Access Control (MAC) layer*, v2, 1996.

[ETS 300 175-4] ETSI, ETS 300 175-4, *Digital Enhanced Cordless Telecommunications (DECT); Common Interface (CI); Part 4: Data Link Control (DLC) layer*, v2, 1996.

[ETS 300 175-5] ETSI, ETS 300 175-5, *Digital Enhanced Cordless Telecommunications (DECT); Common Interface (CI); Part 5: Network (NWK) layer*, v3, 1997.

[ETS 300 175-6] ETSI, ETS 300 175-6, *Digital Enhanced Cordless Telecommunications (DECT); Common Interface (CI); Part 6: Identities and addressing*, v2, 1996.

[ETS 300 175-9] ETSI, ETS 300 175-9, *Digital Enhanced Cordless Telecommunications (DECT); Common Interface (CI); Part 9: Public Access Profile (PAP)*, v2, 1996.

[ETS 300 176-1] ETSI, ETS 300 176-1, *Digital Enhanced Cordless Telecommunications (DECT); Approval test specification; Part 1: Radio*, v2, 1996.

[ETS 300 176-2] ETSI, ETS 300 176-2, *Digital Enhanced Cordless Telecommunications (DECT); Approval test specification; Part 2: Speech*, v2, 1996.

[ETS 300 406] ETSI, ETS 300 406, *Methods for Testing and Specification (MTS); Protocol and profile conformance testing specifications; Standardization methodology*, v1, 1995.

[ETS 300 444] ETSI, ETS 300 444, *Digital Enhanced Cordless Telecommunications (DECT); Generic Access Profile (GAP)*, v1, 1995.

SPICUM servicio de publicaciones

[ETS 300 474-1] ETSI, ETS 300 474-1, *Digital Enhanced Cordless Telecommunications (DECT); Generic Access Profile (GAP); Profile requirement list and profile specific Implementation Conformance Statement (ICS) proforma; Part 1: Portable radio Termination (PT)*, v1, 1996.

[ETS 300 474-2] ETSI, ETS 300 474-2, *Digital Enhanced Cordless Telecommunications (DECT); Generic Access Profile (GAP); Profile requirement list and profile specific Implementation Conformance Statement (ICS) proforma; Part 2: Fixed radio Termination (FT)*, v1, 1996.

[ETS 300 476] ETSI, ETS 300 476, *Digital Enhanced Cordless Telecommunications (DECT); Common Interface (CI); Protocol Implementation Conformance Statement (PICS) proforma*, v1, 1996.

[ETS 300 476-1] ETSI, ETS 300 476-1, *Digital Enhanced Cordless Telecommunications (DECT); Common Interface (CI); Protocol Implementation Conformance Statement (PICS) proforma; Part 1: Network (NWK) layer - Portable radio Termination (PT)*, v1, 1996.

[ETS 300 476-2] ETSI, ETS 300 476-2, *Digital Enhanced Cordless Telecommunications (DECT); Common Interface (CI); Protocol Implementation Conformance Statement (PICS) proforma; Part 2: Data Link Control (DLC) layer - Portable radio Termination (PT)*, v1, 1996.

[ETS 300 476-5] ETSI, ETS 300 476-5, *Digital Enhanced Cordless Telecommunications (DECT); Common Interface (CI); Protocol Implementation Conformance Statement (PICS) proforma; Part 5: Data Link Control (DLC) layer - Fixed radio Termination (FT)*, v1, 1996.

[ETS 300 494] ETSI, ETS 300 494, *Digital Enhanced Cordless Telecommunications (DECT); General Access Profile (GAP); Profile Test Specification (PTS)*, v1, 1996.

[ETS 300 494-1] ETSI, ETS 300 494-1, *Digital Enhanced Cordless Telecommunications (DECT); General Access Profile (GAP); Profile Test Specification (PTS); Part 1: Summary*, v1, 1996.

[ETS 300 494-2] ETSI, ETS 300 494-2, *Digital Enhanced Cordless Telecommunications (DECT); Generic Access Profile (GAP); Profile Test Specification (PTS); Part 2: Profile Specific Test Specification (PSTS) - Portable radio Termination (PT)*, v1, 1996.

[ETS 300 494-3] ETSI, ETS 300 494-3, *Digital Enhanced Cordless Telecommunications (DECT); Generic Access Profile (GAP);Profile Test Specification (PTS); Part 3: Profile Specific Test Specification (PSTS) - Fixed radio Termination (FT)*, v1, 1996.

[ETS 300 550] ETSI, ETS 300 550, *European digital cellular telecommunications system (Phase 2); Mobile Station - Base Station System (MS - BSS) interface; General aspects and principles (GSM 04.01)*, v1, 1994.

SPICUM servicio de publicaciones

[ETSI04] ETSI Mobile Competence Centre, *Overview of 3GPP Release 99 – Summary of all Release 99 Features*, http://www.3gpp.org/ftp/tsg_cn/TSG_CN/TSGN_23/Docs/PDF/NP-040010.pdf, 2004.

[EWOS96] EWOS, *Guide to Open Systems Specification - Conformance, Interoperability and Testing*, 1999.

[EZSDL] *United Computer Scientists*, http://www.ucs.com.pt/ez/, 2004.

[FERN04] M. Fernández-Utrilla, *Desarrollo de Aplicaciones de Localización e Intercambio de Datos sobre Dispositivos Bluetooth*, ETSI Telecomunicación, Universidad de Málaga, 2004.

[FERR00] R. Ferrer, *Interfaz de Operación para Ejecución de Pruebas de Conformidad*, ETSI Telecomunicación, Universidad de Málaga, 2000.

[FISC93] J. Fischer, R. Schröder, *Combined Specification Using SDL and ASN.1*, 6th SDL Forum – Using Objects (SDL'93), pp. 293-304, 1993.

[FSIQ26] Rohde & Schwarz, *Signal Analyzer FSIQ26, Operating manual*, v1119.5063, 2000.

[FUJI94] R. U. Fujii, *Independen Verification and Validation*, Encyclopedia of Software Engineering, editor J. Marciniak, John Wiley & Sons, pp. 568-572, 1994.

[G.726] ITU-T, Recommendation G.726, *40, 32, 24, 16 kbit/s adaptive differential pulsecode modulation (ADPCM)*, 1990.

[GALL00] J. Gallango, *Herramienta Generadora de Código SDL a Partir de MSC*, ETSI Telecomunicación, Universidad de Málaga, 2000.

[GARC00] J. A. García, *Sistema Bluetooth: Software para el Control de la Interface Host-Modulo Bluetooth (HCI)*, ETSI Telecomunicación, Universidad de Málaga, 2000.

[GARC01] A. J. García, *Editor Gráfico de SDL sobre Windows 98*, ETSI Telecomunicación, Universidad de Málaga, 2001.

[GARS01] L. M. Garshol, *BNF and EBNF: What are they and how they work?*, http://www.garshol.priv.no/download/text/bnf.html, 2003.

[GCI96] F. Brady, R. M. Barker, *GCI Interface Specification*, INTOOL GCI/NPL038, NPL, 1996.

[GEPP01] B. Geppert, F. Rö&szlgr, *The SDL pattern approach — a reuse-driven SDL design methodology*, Computer Networks: The International Journal of Computer and Telecommunications Networking archive, vol. 35, nº 6, pp. 627-645, Mayo 2001.

[GIMI05] L. F. Gimilio, *Análisis del Protocolo Mobile IP sobre Redes Bluetooth*, ETSI Telecomunicación, Universidad de Málaga, 2005.

[GLAS93] R. L. Glass, *The Many Flavors of Testing*, Journal of Systems and Software, vol. 20, nº 2, pp. 105-106, 1993.

SPICUM servicio de publicaciones

[GOME01a] J. A. Gómez, J. Poncela, U. Fernández, F. Ruiz, J. T. Entrambasaguas, *Arquitectura genérica para puebas de radio y pruebas de protocolo*, XVI Simposium Nacional de la Unión Científica Internacional de Radio (URSI'01), Villaviciosa de Odón (Madrid), 19-21 Septiembre 2001.

[GOME01b] J. A. Gómez, *Implementación de una Arquitectura Genérica para Control de Pruebas de Protocolo y de Radio*, ETSI Telecomunicación, Universidad de Málaga, 2001.

[GRAH94] D. R. Graham, *Testing*, Encyclopedia of Software Engineering, editor J. Marciniak, John Wiley & Sons, pp. 1330-1353, 1994.

[GUOC97] L. Guochun, *Integrated Modelling & Simulation of Signalling Protocols at B-ISDN UNI on an SDL'92-Based Platform for Performance Eval*uation, Proc. IEEE Symp. On Computers and Communications, pp. 86-90, 1997

[HAL98] F. Halsall, *Comunicación de datos, redes de computadores y sistemas abiertos*, Addison-Wesley Iberoamericana, 4ª ed., ISBN 0-201-65307-9, 1998.

[HALL90] A. Hall, *Seven Myths of Formal Methods*, IEEE Software, vol. 7, nº 5, pp. 11-19, 1990.

[HAUG93] Ø. Haugen, R. Bræk, G. Melby, *The SISU Project*, 6th SDL Forum – Using Objects (SDL'93), pp. 479-489, 1993.

[HEAP01] Hunt Engineering, *HERON-API, Hardware Access Library for Hunt Engineering, C6000 HERON processing modules, User Manual*, v3.3, 2001.

[HERO00] Hunt Engineering, *HERON4, HERON Processing Module for C6201 & C6701, User Manual*, v2.3.c, 2000.

[HETZ88] B. Hetzel, *The Complete Guide to Software Testing*, QED Information Sciences, ISBN 0-89435-242-3,1988.

[HOLU90] A. I. Holub, *Compiler Design in C*, Prentice-Hall, ISBN 0-13-155045-4, 1990.

[ICO00] ICO Global Communications, http://www.ico.com/, 1999.

[ICUM17] Proyecto UMTS, UMA-IC02-33-00017, *MAC: Medidas realizadas en Win2000*, Ingeniería de Comunicaciones, 2002.

[ICUM25] Proyecto UMTS, UMA-IC02-33-00025, *RLC: Medidas de tiempo en Win2000*, Ingeniería de Comunicaciones, 2002.

[ICUM28] Proyecto UMTS, UMA-IC02-33-00028, *RLC: Diseño Control, Multiplexores y Buffers*, Ingeniería de Comunicaciones, 2003.

[ICUM29] Proyecto UMTS, UMA-IC02-33-00029, *RLC: Diseño Modo TM*, Ingeniería de Comunicaciones, 2003.

[ICUM30] Proyecto UMTS, UMA-IC02-33-00030, *MAC: Variables y tipos definidos*, Ingeniería de Comunicaciones, 2004.

SPICUM servicio de publicaciones

[ICUM31] Proyecto UMTS, UMA-IC02-33-00031, *MAC: Elección del formato de transporte*, Ingeniería de Comunicaciones, 2003.

[ICUM32] Proyecto UMTS, UMA-IC02-33-00032, *MAC: Diseño*, Ingeniería de Comunicaciones, 2004.

[ICUM33] Proyecto UMTS, UMA-IC02-33-00033, *RLC: Diseño Modo UM*, Ingeniería de Comunicaciones, 2003.

[ICUM34] Proyecto UMTS, UMA-IC02-33-00034, *RLC: Diseño Modo AM*, Ingeniería de Comunicaciones, 2004.

[ICUM41] Proyecto UMTS, UMA-IC02-33-00041, *Interfaz entre el Sistema SDL y la capa física*, Ingeniería de Comunicaciones, 2003.

[ICUM47] Proyecto UMTS, UMA-IC02-33-00047, *Interfaz del sistema SDL para pruebas internas*, Ingeniería de Comunicaciones, 2004.

[ICUM49] Proyecto UMTS, UMA-IC02-33-00049, *Pruebas de integración*, Ingeniería de Comunicaciones, 2004.

[ICUM94] Proyecto UMTS, UMA-IC02-33-00094, *Generación de Trazas en SDL*, Ingeniería de Comunicaciones, 2004.

[IEEE 488] IEEE, Std 488.1-1987, *IEEE Standard Digital Interface for Programmable Instrumentation –Description*, 1987.

[IEEE 488-2] IEEE, Std 488.2-1992, *IEEE Standard Codes, Formats, Protocols, and Common Commands for Use With IEEE Std 488.1-1987, IEEE Standard Digital Interface for Programmable Instrumentation – Description*, 1992.

[IEEE 802.15-1] IEEE, Std 802.15.1-2005, *IEEE Standard for Information technology--Telecommunications and information exchange between systems-- Local and metropolitan area networks--Specific requirements. Part 15.1: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Wireless Personal Area Networks (WPANs(tm))*, 2005.

[I-ETS 300 176] ETSI, I-ETS 300 176, *Digital Enhanced Cordless Telecommunications (DECT); Approval test specification*, v1, 1992.

[INTE02] Interdigital, *Tau SDL Suite Used To Develop 3G technology*, http://www.3g.co.uk/PR/July2000/3710.htm, 2002.

[IRDA03] Infrared Data Association, *Object Exchange Protocol (OBEX)*, v1.3, 2002.

[ISO 14977] ISO 14977, *Information technology -- Syntactic metalanguage -- Extended BNF*, 1996.

[ISO 7498] ISO 7498, *Information technology -- Open Systems Interconnection -- Basic Reference Model*, 1989/97.

[ISO 9646] ISO 9646, *Information technology -- Open Systems Interconnection -- Conformance testing methodology and framework*, 1994/98.

[ITCO02] ITCom, *Comparison of Wireless Technologies*, http://www.itcom.itd.umich.edu/wireless/options.html, 2003.

SPICUM servicio de publicaciones

[ITUTX] ITU-T, *Recommendations Series X, Data networks, open system communications and security*, http://www.itu.int/rec/T-REC-X/e.

[ITUTZ] ITU-T, *Recommendations Series X, Languages and general software aspects for telecommunication systems*, http://www.itu.int/rec/T-REC-Z/e.

[IZQU03] A. Izquierdo, *Validación de Sistemas Móviles de 3ª Generación Modelados en SDL*, ETSI Telecomunicación, Universidad de Málaga, 2003.

[JACO92] I. Jacobson, et al., *Object-Oriented Software Engineering: A Use Case-Driven Approach*, Addison-Wesley. ISBN 0-201-54435-0, 1994.

[JAVA] Sun Microsystems, *Java Technology*, http://java.sun.com/, 2000.

[JDK] Sun Microsystems, *Java 2 Platform, Standard Edition (J2SE)*, http://java.sun.com/products/jdk/1.2/, 2000.

[JOHN95] S. C. Johnson, *Yacc - Yet Another Compiler-Compiler*, http://dinosaur.compilertools.net/yacc/index.html, 1995.

[KARN93] G. Karner, *Semantic Integration of ASN.1 into SDL*, 6th SDL Forum – Using Objects (SDL'93), pp. 305-315, 1993.

[KATS91] K. Katsuyama, F. Sato, T. Nakakawaji, T. Mizuno, *Strategic Testing Environment with Formal Description Techniques*, IEEE Transactions on Computers, vol. 40, nº 4, 1991.

[KAZA95] K. Kazama, S. Suzuki, M. Hatafuku, *Design and Implementation of Interconnectability Testing System*, IEICE Transactions on Communications, Vol. E78-B, nº 3, 1995.

[KERB99] A. Kerbrat, T. Jéron, R. Groz, *Automated Test Generation from SDL Specifications*, 9th SDL Forum – The Next Millenium (SDL'99), pp. 135-151, Montreal (Canada), Junio 1999.

[KERN03] *The Linux Kernel Archives*, http://www.kernel.org, 2003.

[KEUM98] C. S. Keum, J. K. Lee, D. G. Lee, B. S. Lee, *Integrated environment based on object-oriented methodology for real-time systems*, First International Symposium on Object-Oriented Real-Time Distributed Computing (ISORC98), pp. 284-288, 20-22 Abril 1998.

[KIM01] Y. Kim, H. Jung, H. H. Lee, K. R. Cho, *MAC Implementation for IEEE 802.11 Wireless LAN*, Joint 4th IEEE Int. Conf. On ATM (ICATM 2001) and High Speed Intelligent Internet Symposium, pp. 191-195, 2001.

[KOVA06] G. Kovács, Z. Pap, G. Csopaki, K. Tarnay, *Iterative automatic test generation method for telecommunication protocols*, Computer Standards & Interfaces, vol. 28, nº 4, pp. 412-427, 2006.

[LAPE88] F. J. De Lapeyre, K. C. Woolaard, S. Yvergnaux, *The Use of an ISO Formal Technique for Conformance Test Specifications*, BT Telecom Technology Journal, vol. 6, nº 1, pp. 23-30, 1988.

SPICUM servicio de publicaciones

[LARA04] M. Lara, *Implementación del Perfil HID de Bluetooth*, ETSI Telecomunicación, Universidad de Málaga, 2004.

[LEEA06] J. D. Leea, J. I. Jungb, J. H. Leec, J. G. Hwangc, J. H. Hwangd, S. U. Kim, *Verification And Conformance Test Generation Of Communication Protocol For Railway Signaling Systems*, Computer Standards & Interfaces, vol. 29, nº 2, pp. 143-151, 2007.

[LEONAR] *Da Vinci Communications Ltd.*, http://www.davinci-communications.com/company.shtml, 2004.

[LESK75] M. E. Lesk, E. Schmidt, *Lex - A Lexical Analyzer Generator*, http://dinosaur.compilertools.net/lex/index.html, 1975.

[LEVI92] J. Levine, T. Mason, D. Brown, Lex & Yacc, O'Reilly, ISBN 1-565-92000-7, 1992.

[LINA01] A. M. Linares, *Implementación de un Parser para Lenguaje TTCN*, ETSI Telecomunicación, Málaga, 2001.

[LINN89] R. J. Linn, *Conformance evaluation methodology and protocol testing*, IEEE Journal on Selected Areas in Communications, vol. 7, nº 7, pp. 1143-1158, 1989.

[LLAB00] F. Llabrés, *Sistema Bluetooth: Implementación de la Capa de Gestión de Enlaces*, ETSI Telecomunicación, Universidad de Málaga, 2000.

[LLOY89] M. Lloyd, *Conformance: Assuring Interoperability*, Computer Networks & ISDN Systems, vol. 17, pp. 367-370, 1989.

[LOBA02] S. Lobato, *Formalización de las Pruebas Radio para Equipos Bluetooth*, ETSI Telecomunicación, Universidad de Málaga, 2002.

[LOBA08] S. Lobato, E. Ruiz, J. A. Vigo, J. M. Jiménez, *Generalised Architecture for Radio Conformance Testers using TTCN-3*, The TTCN-3 User Conference, Madrid, 2008.

[LUQI97] Luqi, J. A. Goguen, *Formal Methods: Promises and Problems*, IEEE Software, vol. 14, nº 1, pp. 73-85, 1997.

[MART03] A. Martinez, *Herramienta de Edición para Lenguaje TTCN*, ETSI Telecomunicación, Universidad de Málaga, 2003.

[MDA03] Object Management Group, *Model Driven Architecture (MDA) Guide*, v1.0.1, 2003.

[METT99] R. Mettala, *Bluetooth Protocol Architecture v1.0*, Bluetooth White Paper, Agosto 1999.

[MINT] AT4 wireless, *Mobile Communications Integrated Tester (MINT)*, http://www.at4wireless.com/web_esp/sistemas/soluciones_ensayo/mint, 2007.

[MITC93] L. Mitchell, S. Lu, *Specifications and Validations of Inmarsat Aeronautical System Protocols*, 6th SDL Forum – Using Objects (SDL'93), pp. 51-63, 1993.

SPICUM servicio de publicaciones

[MORA99] C. Morales, *Sistema ICO: Cobertura Universal Vía Satélite*, Telefonía & Comunicaciones, pp. 66-68, 1999.

[MORE95] P. Morel, *Mobility in MAP Networks Using the DECT Wireless Protocols*, Proc. IEEE Int. Workshop on Factory Communication Systems (WFCS'95), pp. 145-149, 1995.

[MORI02a] M. V. Morillo-Velarde, L. Sánchez, B. Soret, J. Torrecilla, J. Poncela, *Diseño e Implementación de Sistemas de Prueba para la Certificación de Equipos Bluetooth,* 2º Congreso Internacional de Telemática (CITEL02), La Habana (Cuba), 25-29 Noviembre 2002.

[MORI02b] M. V. Morillo-Velarde, *Desarrollo de un Sistema de Pruebas de Interoperatividad para el Perfil Headset de Bluetooth*, ETSI Telecomunicación, Universidad de Málaga, 2002.

[MORI03] M. V. Morillo-Velarde, J. Torrecilla, J. Poncela, *Sistema de Pruebas para el Perfil Headset de Bluetooth*, XVIII Simposium Nacional de la Unión Científica Internacional de Radio (URSI'03), A Coruña, 10-12 Septiembre 2003.

[MORI04] M. V. Morillo-Velarde, J. F. Colás, J. Poncela, B. Soret, J. T. Entrambasaguas, *UMTS Protocol Development Using Formal Languages*, Communication Systems and Networks (CSN 2004), Marbella (España), 1-3 Septiembre 2004.

[MTS] ETSI, *Comité ETSI, Methods for Testing & Specification*, http://portal.etsi.org/portal_common/home.asp?tbkey1=MTS, 2004.

[MTS96] MTS, *Methods for Testing and Specification*, 1996.

[NAIK92] K. Naik, B. Sarikaya, *Testing Communication Protocols*, IEEE Software, vol. 9, nº 1, pp. 27-37, 1992.

[NATI01] National Semiconductor, *Application Areas of DECT systems*, http://www.national.com/appinfo/wireless/files/ApplicationArea.pdf , 2001.

[NELS96] J. Nelson, M. Barry, G. Fleming, D. Madden, *UMTS Signalling Network Layer and its Verification Using SDL*, Proc. of the ACTS Mobile Telecommunications Summit, pp. 74-84, 1996.

[NOER96] A. R. Noerpel, Y. B. Lin, H. Sherry, *PACS: Personal Access Communications System – A Tutorial*, IEEE Personal Communications, vol. 3, nº 3, pp. 32-43, 1996.

[OLSE94] A. Olsen, O. Færgemand, B. Møller-Pedersen, R. Reed, J. R. W. Smith, *Systems Engineering Using SDL-92*, North-Holland, ISBN 0-44-489872-7, 1994

[OPENTT] *OpenTTCN Tester*, http://www.openttcn.com/, 2004.

[OSTR94] T. J. Ostrand, *Categories of Testing*, Encyclopedia of Software Engineering, editor J. Marciniak, John Wiley & Sons, pp. 90-93, 1994.

SPICUM servicio de publicaciones

[PADG95] J. E. Padgett, C. G. Günther, T. Hattori, Overview of Wireless Personal Communications, IEEE Communications Magazine, vol. 33, nº 1, pp. 28-41, 1995.

[PAXS95] V. Paxson, *Flex - A Fast Scanner Generator*, http://dinosaur.compilertools.net/flex/index.html, 1995.

[PEJA02] M. Pejanovic, *Optimizing IMT-2000 Deployment Strategies*, http://www.itu.int/ITU-D/tech/imt-2000/moscow2002/pejanovic.pdf, Moscú, 2002.

[PINS90] L. J. Pinson, R. S. Wiener, *Applications of object-oriented programming*, Addison-Wesley Publishing Company, ISBN 6-001-15365-5, 1990.

[PLAZ06] A. Plaza, *Integración de Sistemas de Pruebas en DSPs*, ETSI Telecomunicación, Universidad de Málaga, 2006.

[PONC99] J. Poncela, J. T. Entrambasaguas, *Metodología para el Desarrollo de Bancos de Pruebas para Sistemas de Comunicaciones*, II Jornadas de Ingeniería Telemática (JITEL'99), 1999.

[PONC00] J. Poncela, R. Sánchez, P. Tapia, R. Ferrer, J. T. Entrambasaguas, *Testbed Development for Communication Systems Using Formal Languages*, 2nd Workshop of the SDL Forum Society on SDL and MSC (SAM2000), Grenoble (Francia), 26-28 Junio 2000.

[PONC07a] J. Poncela, J. Gómez, C. Valero, U. Fernández, *Using TTCN for Radio Conformance Test Systems*, 13th System Design Language Forum (SDL Forum'07), París (Francia), 18-21 Septiembre 2007.

[PONC07b] J. Poncela, J. Gómez, S. Lobato, F. Ruiz, *Implementation of Test Systems for Radio Equipment with TTCN*, XV Jornadas de Concurrencia y Sistemas Distribuidos (JCSD'07), Torremolinos (Málaga), 6-8 Junio 2007.

[POSI99] IEEE, Std 1003.1, *IEEE Standard for Information Technology - Portable Operating System Interface (POSIX)*, 1999.

[POST94] R. M. Poston, *Automating Testing from Object Models*, Communications of the ACM, vol. 37, nº 9, pp. 48-58, 1994.

[PRAS05] M. Prasanna, S. N. Sivanandam, R. Venkatesan, R. Sundarrajan, *A Survey On Automatic Test Case Generation*, Academic Open Internet Journal, vol. 15, 2005.

[PRIX02] Alcatel-Lucent, *The 60th Monaco Grand Prix uses Alcatel's mobility solutions*, http://www.alcatel.at/at/presse/archiv/pressearchiv2002/content/01809/, Mayo 2002.

[PROB89] R. L. Probert, H. Ural, M. W. A. Hornbeek, *A Comprehensive Software Environment for Developing Standardized Conformance Test Suites*, Computer Networks and ISDN Systems, vol. 18, nº 1, pp. 19-29, 1989.

SPICUM servicio de publicaciones

[PROY95] Proyecto I+D, *Analizador de Protocolos para Sistemas DECT*, Junta de Andalucía. Consejería de Economía y Hacienda, Convenio de Cooperación en Materia de Innovación Industrial y Tecnología con la Universidad de Málaga, C. Camacho, CETECOM, S.A., Junio 1995 – Mayo 1996.

[PROY99] Proyecto I+D, *Herramientas para pruebas de sistemas y redes de telecomunicación*, Unión Europea, Fondos FEDER (1FD97-0650), J. T. Entrambasaguas, Dpto. Ingeniería de Comunicaciones, Junio 1999 – Diciembre 2001.

[PROY02] Proyecto I+D, *Comunicaciones móviles de tercera generación*, Plan Nacional de I+D+I, Programa Nacional de Tecnologías de la Información y de las Telecomunicaciones (FIT-070000-2002-51), A. Moreno, J. T. Entrambasaguas, CETECOM S.A. / Departamento de Ingeniería de Comunicaciones, Enero 2002 – Diciembre 2002.

[PROY03] Proyecto I+D, *Comunicaciones móviles de tercera generación*, A. Moreno, J. T. Entrambasaguas, Plan Nacional de I+D+I, Programa Nacional de Tecnologías de la Información y de las Telecomunicaciones (FIT-070000-2003-72), CETECOM S.A./ Departamento de Ingeniería de Comunicaciones, Enero 2003 – Diciembre 2003.

[Q.1214] ITU-T, Recommendation Q.1214, *Distributed functional plane for intelligent network CS1*, 1993.

[Q.920] ITU-T, Recommendation Q.920, *ISDN user-network interface data link layer - General aspects*, 1993.

[Q.921] ITU-T, Recommendation Q.921, *ISDN user-network interface - Data link layer specification*, 1997

[Q.931] ITU-T, Recommendation Q.931, *ISDN user-network interface layer 3 specification for basic call control*, 1998.

[QIXI96] P. Qixiang, C. Shiduan, J. Yuehui, *Protocol Conformance Test Suite Generation*, Proceedings of ICCT'96, International Conference on Communication Technology, vol.1, pp. 218-222, 1996.

[REED96] R. Reed, *SDL+: Methodology for real time systems*, Computer Networks and ISDN Systems, nº 28, pp. 1685- 1701, 1996.

[REED01] R. Reed, *Notes on SDL for the New Millennium*, Computer Networks, vol. 35, pp. 709-720, 2001.

[REIS95] M. Reiss, *Radio: an Alternative for the Loop*, Telcom Report International, 1995.

[RELE99] 3GPP, *Specifications - Releases (and phases and stages)*, http://www.3gpp.org/specs/releases.htm, 2005.

[RIO04] M. Rio, M. Goutelle, T. Kelly, R. Hughes-Jones, J. P. Martin-Flatin, Y. T. Li, *A Map of the Networking Code in Linux Kernel 2.4.20*, Technical Report DataTAG-2004-1 FP5/IST DataTAG Project, 2004.

SPICUM servicio de publicaciones

[RODR99] M. Rodríguez, R. Calmeau, E. Fernández, *Application of SDL-92 for the specification of OSI Management Systems*, Proc. 6th IFIP/IEEE Int. Symp. On Integrated Network Management, pp. 447-460, 1999.

[RODR04] J. M. Rodríguez, *Modelado SDL de Funciones Avanzadas para Sistemas Móviles de Tercera Generación*, ETSI Telecomunicación, Universidad de Málaga, 2004.

[RODR07] M. Rodríguez, J. M. Parra, *Experiences in Using the SOMT Method to Support the Design and Implementation of Network Simulator*, 13th System Design Language Forum (SDL Forum'07), París (Francia), 18-21 Septiembre 2007.

[ROHDE] *Rohde & Schwarz*, http://www.rohde-schwarz.com/, 2004.

[ROHDUM] Rohde & Schwarz, *CRTU Protocol Test Platform*, http://www2.rohde-schwarz.com/en/products/test_and_measurement/product_categories/mobile_radio/CRTU.html, 2007.

[ROME03] A. Romero, *Aplicaciones Bluetooth sobre Obex*, ETSI Telecomunicación, Universidad de Málaga, 2003.

[RTDEVE] *PragmaDev*, http://www.pragmadev.com/, 2004.

[RUIZ06] C. Ruiz, *Modelado de Redes Inalámbricas en SDL*, ETSI Telecomunicación, Universidad de Málaga, 2006.

[RUMB91] J. Rumbaugh, M. Blaha, W. Premerlani, F. Eddy, W. Lorensen, *Object-Oriented Modeling and Design*, Prentice Hall. ISBN 0-13-629841-9, 1991.

[SAFIRE] *Solinet GmbH*, http://www.safire-sdl.com/, 2004.

[SALM98] L. Salmerón, *Enlace de los Casos de Prueba de la TBR22 con el Software de un Sistema para Análisis de Protocolo de Terminales DECT*, ETSI Telecomunicación, Universidad de Málaga, 1998.

[SANC00a] R. A. Sánchez, *Desarrollo de un Equipo Automático de Pruebas de DECT*, ETSI Telecomunicación, Universidad de Málaga, 2000.

[SANC00b] R. Sánchez, *Sistema Bluetooth: Control de Enlace Lógico y Protocolo de Adaptación (L2CAP)*, ETSI Telecomunicación, Universidad de Málaga, 2000.

[SANC02] L. Sánchez, *Banco de Pruebas para la Capa de Gestión de Enlace LM del Sistema Bluetooth*, ETSI Telecomunicación, Málaga, 2002.

[SANG97] K. Sangki, K. Sungun, *INAP Protocol Conformance Testing*, Proc. Intelligent Network Workshop, vol. 2, 4-7 Mayo1997.

[SARI89] B. Sarikaya, *Conformance Testing: Architectures and Test Sequences*, Computer Networks and ISDN Systems, vol. 17, nº 2, pp. 111-126, 1989.

[SCHM00] M. Schmitt, M. Ebner, J. Grabowski, *Test Generation with Autolink and TestComposer*, Proc. 2nd Workshop of the SDL Forum Society on SDL and MSC, 2000.

SPICUM servicio de publicaciones

[SCPI99] SCPI Consortium, *Standard Commands for Programmable Instrumentation (SCPI)*, vol.1, 1999.

[SERR06] C. Serrano, *Sistema de Posicionamiento Local Basado en Tecnología Bluetooth*, ETSI Telecomunicación, Universidad de Málaga, 2006.

[SG17] ITU-T, Study Group 17, http://www.itu.int/ITU-T/studygroups/com17/index.asp, 2001.

[SIDH93] D. Sidhu; A. Chung; H. Motteler; T. Blumer; Y. Yesha, *Formal Methods in Protocol Development*, Journal of Computer and Software Engineering, vol. 1, nº 3, pp. 257-279, 1993.

[SIEB97] G. Siebert, A. Sharma, *ATM via SDL*, http://www.telelogic.se.techpapper/cellware/page1.htm, 1997.

[SIPI01] J. Sipilä, V. Luukkala, *An SDL Implementation Framework for Third Generation Mobile Communications System*, Proceedings of the 10th International SDL Forum Copenhagen on Meeting UML, Lecture Notes In Computer Science, vol. 2078, 2001.

[SIT94a] SiTel Sierra, *SC14401 DECT Processor Evaluation System User Manual*, v2.0, Noviembre 1994.

[SIT94b] SiTel Sierra, *SC144xx DECT Family Commands manual*, v5.0, Noviembre 1994.

[SIT94c] SiTel Sierra, *MAC-CELL Site Functions Software Manual*, v2.0, Noviembre 1994.

[SMIQ03B] Rohde & Schwarz, *Vector Signal Generator SMIQ03B, Operating manual*, v1125.5610, 2001.

[SONN98] D. Sönnerstam, *Comprehensive Description of the Bluetooth System*, Ericsson, Junio 1998.

[SORE02] B. Soret, *Desarrollo de un Banco de Pruebas para el Perfil SPP de Bluetooth*, ETSI Telecomunicación, Universidad de Málaga, 2002.

[SORE03a] B. Soret, U. Fernández, J. Poncela, *Banco de Pruebas para el Perfil SPP de Bluetooth*, XVIII Simposium Nacional de la Unión Científica Internacional de Radio (URSI'03), A Coruña (España), 10-12 Septiembre 2003.

[SORE03b] B. Soret, M. V. Morillo-Velarde, J. F. Colás, J. Poncela, *Modelado en SDL de la Capa 2 de UMTS*, IV Jornadas de Ingeniería Telemática (JITEL'03), Gran Canaria (España), 15-17 Septiembre 2003.

[SPEC93] R. Reed et al. (Eds.), *The SPECS Consortium*, SPECS, North-Holland, ISBN 0-444-89923-5,1993.

[STAM97] R. Stamvik, *Integrating SDT Generated Code with Target Platforms*, http://www.telelogic.se/techpapper/sdtgen.htm, 1997.

[SWING] Sun Microsystems, *The Swing Tutorial - Creating a GUI with JFC Swing*, http://java.sun.com/docs/books/tutorial/uiswing/, 2000.

SPICUM servicio de publicaciones

[TAE07] T. H. Kim, *SDL Design and Performance Evaluation of a Mobility Management Technique for 3GPP LTE systems*, 13th SDL Forum, Paris (Francia), 18-21 Septiembre 2007.

[TAPI00] P. Tapia, *Desarrollo de un Banco de Pruebas para Telefonía Inalánbrica DECT*, ETSI Telecomunicación, Universidad de Málaga, 2000.

[TAU] *Telelogic AB*, Telelogic Tau, http://www.telelogic.com/products/tau/index.cfm, 1997.

[TAYL92] D. A. Taylor, *Object-oriented information systems: planning and implementation*, John Wiley & Sons, ISBN 0-471-54364-0,1992.

[TBR 006] ETSI, TBR 006, *Digital Enhanced Cordless Telecommunications (DECT); General terminal attachment requirements*, v3, 1999.

[TBR 010] ETSI, TBR 010, *Digital Enhanced Cordless Telecommunications (DECT); General Terminal Attachment Requirements; Telephony Applications*, v3, 1999.

[TBR 022] ETSI, TBR 022, *Radio Equipment and Systems (RES); Attachment requirements for terminal equipment for Digital Enhanced Cordless Telecommunications (DECT) Generic Access Profile (GAP) applications*, v1, 1997.

[TBR 036] ETSI, TBR 036, *Digital Enhanced Cordless Telecommunications (DECT); Global System for Mobile communications (GSM); DECT access to GSM Public Land Mobile Networks (PLMNs) for 3,1 kHz speech applications*, v1, 1998.

[TBR 040] ETSI, TBR 040, *Digital Enhanced Cordless Telecommunications (DECT); Integrated Services Digital Network (ISDN); Attachment requirements for terminal equipment for DECT/ISDN interworking profile applications,* v1, 1998.

[TERN03] J. Ternero, *Implementación en SDL del Protocolo RFCOMM*, ETSI Telecomunicación, Universidad de Málaga, 2003.

[TIBIOS] Texas Instruments, *TMS320C6000 DSP/BIOS Application Programming Interface (API) Reference Guide*, 2002.

[TIC67] Texas Instruments, *TMS320C6000 CPU and Instruction Set Reference Guide*, 1999.

[TIEVM] Texas Instruments, *TMS320C6201/6701 Evaluation Module User´s Guide*, 1998.

[TOOL07] Tool, http://en.wikipedia.org/wiki/Tool, 2007.

[TORR02] J. F. Torreblanca, *Multiplexación de Canales de Transporte y Entramado de Canales Físicos en el Modo FDD de UMTS*, ETSI Telecomunicación, Universidad de Málaga, 2002.

[TR 102 185] ETSI, TR 102 185, *Digital Enhanced Cordless Telecommunications (DECT); Data Services Profile (DSP); Profile overview*, v1.2.1, 2001.

SPICUM servicio de publicaciones

[TR 102 570] ETSI, TR 102 570, *Digital Enhanced Cordless Telecommunications (DECT); New Generation DECT;Overview and Requirements*, v1.1.1, 2007.

[TREM85] J. P. Tremblay, P. G. Sorenson, *The Theory and Practice of Compiler Writing*, Mc Graw-Hill, ISBN 0-07-065161-2, 1985.

[TRET94] J. Tretmans, *A Formal Approach to Conformance Testing*, Protocol Test Systems, VI, IFIP, pp. 257-276, 1994.

[TS 07.10] Ver [TS 101 369].

[TS 101 369] ETSI, TS 101 369, *Digital cellular telecommunications system (Phase 2+); Terminal Equipment to Mobile Station (TE-MS) multiplexer protocol (3PP TS 07.10 version 6.5.0 Release 1997)*, v6.5.0, 2002.

[TS 101 863] ETSI, TS 101 863, *Digital Enhanced Cordless Telecommunications (DECT); DECT/UMTS Interworking Profile (IWP)*, v1.1.2, 2001.

[TS 102 012] ETSI, TS 102 012, *Digital Enhanced Cordless Telecommunications (DECT); DECT Packet Radio Service (DPRS); Application Specific Access Profile (ASAP): V.24 Interworking; Profile Test Specification (PTS)*, v1.1.1, 2001.

[TS 102 014] ETSI, TS 102 014, *Digital Enhanced Cordless Telecommunications (DECT); DECT Packet Radio Service (DPRS); Application Specific Access Profile (ASAP): Ethernet Interworking; Profile Test Specification (PTS)*, v1.1.1, 2001.

[TS 102 265] ETSI, TS 102 265, *Digital Enhanced Cordless Telecommunications (DECT); DECT access to IP networks*, v1.2.1, 2004.

[TS 102 342] ETSI, TS 102 342, *Digital Enhanced Cordless Telecommunications (DECT); Cordless multimedia communication system; Open Data Access Profile (ODAP)*, v1.2.1, 2005.

[TS 102 379] ETSI, TS 102 379, *Digital Enhanced Cordless Telecommunications (DECT); Fixed network Multimedia Message Service (F-MMS) Interworking Profile*, v1.1.1, 2005.

[TS 102 527] ETSI, TS 102 527, *Digital Enhanced Cordless Telecommunications (DECT); New Generation DECT*, v1.1.1, 2007/08.

[TTHREE] *Testing Technologies IST GmbH*, http://www.testingtech.de/, 2004.

[TUTEB] Tutorial One: EBNF, http://www.ugrad.cs.ubc.ca/~cs126/Homepage/tutorials/tutOne.html , 2002.

[TUTT92] W. H. W. Tuttlebee, *Cordless Personal Communications*, IEEE Communications Magazine, vol. 30, nº 12, pp. 42-54, 1992.

[ULTEDT] *UltraEdit*, http://www.ultraedit.com/, 2000.

[UML07] Object Management Group, *Unified Modeling Language*, 2007.

SPICUM servicio de publicaciones

[UYAR90] M. Ü. Uyar, A. Lapone, K. K. Sabnani, *Algorithmic Verification of ISDN Network Layer Protocol*, AT&T Technical Journal, vol. 69, nº 1, pp. 17-31, 1990.

[V.250] ITU-T, Recommendation V.250, *Serial asynchronous automatic dialling and control*, 1999.

[VALE02] C. Valero, *Implementación de una Arquitectura Genérica para Control de Pruebas de Protocolo y Pruebas de Radio y Protocolo para Equipos UMTS*, ETSI Telecomunicación, Universidad de Málaga, 2002.

[VALL00] F. Valle, *Automatización de Pruebas de Sistemas de Comunicaciones Móviles*, ETSI Telecomunicación, Universidad de Málaga, 2000.

[VERI96] Verilog, *ObjectGEODE – Method Guidelines*, 1996.

[VERM94] G. S. Vermeer, H. Blik, *Interoperability Testing: Basis for the Acceptance of Communication Systems*, Protocol Test Systems, VI, IFIP, pp. 315-330, 1994.

[VIGO04] J. A. Vigo, *Construcción de una Plataforma Software para Soporte de Aplicaciones Bluetooth*, ETSI Telecomunicación, Universidad de Málaga, 2004.

[VOLL94] A. Vollmer, *DECT Conquers European Cordless Market*, Electronics, vol. 11, nº 4, pp. 12, 1994.

[WETS06] ETSI, *Test Specifications*, http://portal.etsi.org/mbs/Testing/conformance/conformance.asp.

[WEZE91] C. D. Wezeman, S. Batley, J. A. Lynch, *Formal Methods to Assist Conformance Testing*, Formal Description Techniques III, IFIP, pp. 157-174, Elsevier Science, 1991.

[WHIT94] L. J. White, H. K. N. Leung, *Integration Testing*, Encyclopedia of Software Engineering, editor J. Marciniak, John Wiley & Sons, pp. 573-577, 1994.

[WING94] J. M. Wing, *Formal Methods*, Encyclopedia of Software Engineering, editor J. Marciniak, John Wiley & Sons, pp. 1330-1353, 1994.

[WITA95] D. Witaszek, E. Holz, M. Wasowski, S. Lau, J. Fischer, *A Development Method for SDL-92 Specifications Based on OMT*, Proc. of the 7th SDL Forum, Oslo, 1995.

[WLL600] Urmet Group, *WLL 600: the wireless local loop*, http://www.urmet.it/database.asp?prodotto=62, 2002.

[WRC00] ITU, *World Radiocommunication Conference 2000*, http://www.itu.int/newsarchive/wrc2000/, 2000.

[X.210] ITU-T, Recommendation X.210, *Information technology - Open Systems Interconnection - Basic Reference Model: Conventions for the definition of OSI services*, 1993.

SPICUM servicio de publicaciones

[X.290] ITU-T, Recommendation X.290, *OSI conformance testing methodology and framework for protocol Recommendations for ITU-T applications - General concepts*, 1995.

[X.291] ITU-T, Recommendation X.291, *OSI conformance testing methodology and framework for protocol Recommendations for ITU-T applications - Abstract test suite specification*, 1995.

[X.292] ITU-T, Recommendation X.292, *OSI conformance testing methodology and framework for protocol Recommendations for ITU-T applications - The Tree and Tabular Combined Notation (TTCN)*, 2002.

[X.293] ITU-T, Recommendation X.292, *OSI conformance testing methodology and framework for protocol Recommendations for ITU-T applications - Test realization*, 2005.

[X.294] ITU-T, Recommendation X.294, *OSI conformance testing methodology and framework for protocol Recommendations for ITU-T applications - Requirements on test laboratories and clients for the conformance assessment process*, 1995.

[X.295] ITU-T, Recommendation X.295, *OSI conformance testing methodology and framework for protocol Recommendations for ITU-T applications - Protocol profile test specification*, 1995.

[X.296] ITU-T, Recommendation X.296, *OSI conformance testing methodology and framework for protocol Recommendations for ITU-T applications - Implementation conformance statements*, 1995.

[X.680] ITU-T, Recommendation X.680, *Information technology - Abstract Syntax Notation One (ASN.1): Specification of basic notation*, 2002.

[X.683] ITU-T, Recommendation X.683, *Information technology - Abstract Syntax Notation One (ASN.1): Parameterization of ASN.1 specifications*, 2002.

[X.690] ITU-T, Recommendation X.690, *Information technology - ASN.1 encoding rules: Specification of Basic Encoding Rules (BER), Canonical Encoding Rules (CER) and Distinguished Encoding Rules (DER)*, 2002.

[X.691] ITU-T, Recommendation X.691, *Information technology - ASN.1 encoding rules - Specification of Packed Encoding Rules (PER)*, 2002.

[XILINX] *Xilinx Inc.*, http://www.xilinx.com, 2006.

[YEWE00] W. Ye, Y. Du, W. Deng, *Application of object-oriented analysis and design in CDMA BSS development*, 36th International Conference on Technology of Object-Oriented Languages and Systems (TOOLS - Asia), 2000.

[Z.100] ITU-T, Recommendation Z.100, *Specification and Description Language (SDL)*, 1992.

SPICUM servicio de publicaciones

[Z.105] ITU-T, Recommendation Z.105, *SDL combined with ASN.1 modules (SDL/ASN.1)*, 2003.

[Z.107] ITU-T, Recommendation Z.107, SDL with embedded ASN.1, 1999.

[Z.120] ITU-T, Recommendation Z.120, *Message sequence chart (MSC)*, 1999.

[Z.130] ITU-T, Recommendation Z.130, *ITU object definition language*, 1999.

[Z.150] ITU-T, Recommendation Z.150, *User Requirements Notation (URN) - Language requirements and framework*, 2003.

[Z.200] ITU-T, Recommendation Z.200, *CHILL - The ITU-T Programming Language*, 1999.

[ZENG89] H. Zeng, S. T. Chanson, B. R. Smith, *On Ferry Clip Approaches in Protocol Testing*, Computer Networks and ISDN Systems, vol. 17, nº 2, pp. 77-88, 1989.

SPICUM servicio de publicaciones

APÉNDICES

SPICUM
servicio de publicaciones

SPICUM
servicio de publicaciones

Apéndice

A

APÉNDICE A: SÍMBOLOS SDL

La Tabla A.1 muestra los símbolos empleados en la representación gráfica del lenguaje SDL [Z.100].

Tabla A.1: Símbolos utilizados en diagramas SDL.

Símbolo	Nombre	Símbolo	Nombre
	Arranque (proceso)		Llamada a procedimiento
	Asociación sólida		Marco
	Asociación punteada		Operador
	Bloque/Sistema		Parada
	Clase		Procedimiento
	Comentario		Proceso
	Condición de habilitación o Señal continua		Puerta
	Conector de entrada/salida		Recepción
	Creación de proceso		Referencia de texto
	Crear proceso		Reserva
	Decisión		Ruta/Canal
	Entrada prioritaria		Servicio

SPICUM servicio de publicaciones

Símbolo	Nombre	Símbolo	Nombre
	Envío		Subestructura
	Estado		Tarea
	Extensión de texto		Texto
	Fin de macro		Tipo de bloque
	Fin de procedimiento		Tipo de proceso
	Inicio de macro		Tipo de servicio
	Inicio de procedimiento		Tipo de sistema
	Lista de señales		Transición
	Llamada a macro		

SPICUM servicio de publicaciones

Apéndice

B

APÉNDICE B: DOCUMENTOS DEL COMITÉ MTS

ETSI, como organismo europeo dedicado a la producción de estándares de sistemas de comunicaciones, ha estado notablemente activo en el área de las pruebas de conformidad. En su estructura organizativa existe el Comité Técnico para la definición de Métodos de Prueba y Especificación (TC MTS – *Technical Committee Methods for Testing & Specification*). Este comité es responsable de estudiar y proponer métodos para llevar a cabo las pruebas de certificación de equipos de telecomunicación y para especificar dichos sistemas en los estándares publicados por ETSI[1].

Tabla B.1: Códigos de documentos ETSI.

	Sigla	Nombre
Desde 1998	EN	*European Standard*
	ES	*ETSI Standard*
	EG	*ETSI Guide*
	TS	*ETSI Technical Specification*
	TR	*ETSI Technical Report*
	SR	*Special Report*
Hasta 1998	ETS	*European Telecommunication Standard*
	I-ETS	*Interim European Telecommunication Standard*
	TBR	*Technical Basis for Regulation*
	ETR	*ETSI Technical Report*
	GSM-TS	*GSM Technical Specification*
	TCTR	*Technical Committee Technical Report*
	TCRTR	*Technical Committee Reference Technical Report*

Categoría: + (arriba) – (abajo)

Categoría: + (arriba) – (abajo)

[1] Parte del trabajo realizado por este comité ha sido llevado a cabo por el Centro de Competencia de Protocolos y Pruebas (PTCC – *Protocol and Testing Competence Centre*). Actualmente se ha integrado en el comité MTS.

SPICUM servicio de publicaciones

El comité MTS ha generado a lo largo de su existencia un conjunto de documentos que ayudan en el proceso de desarrollo de las normas ETSI, aunque también son de aplicación fuera de este organismo. Este comité centra sus esfuerzos en el uso de lenguajes y notaciones formales estandarizadas (SDL, ASN.1, MSC, TTCN, UML), colaborando también en el desarrollo y mantenimiento de los mismos; por ejemplo, la notación TTCN-3.

Por el interés que supone este trabajo se recogen a continuación los documentos publicados por este comité, tanto aquellos que poseen la categoría de estándares como los informes técnicos que ha elaborado. La identificación de cada uno viene dada por las primeras letras del código del documento; este identificador fue modificado en 1998. En la Tabla B.1 se recogen cada uno de los códigos posibles tanto en la nueva versión como en la antigua. La explicación de cada uno de los tipos de documentos que existen se puede conseguir en [DELI01] y [DELI02]. La Tabla B.2 recoge los estándares publicados, mientras que la Tabla B.3 recoge los informes y directivas.

Tabla B.2: Lista de estándares producidos por TC MTS.

Código	Título
ES 201 770	Title: Methods for Testing and Specification (MTS);Test synchronization architectural reference;Test synchronization protocol 1plus (TSP1+) specification
ES 201 873-1	Title: Methods for Testing and Specification (MTS);The Tree and Tabular Combined Notation version 3;Part 1: TTCN-3 Core Language
ES 201 873-2	Title: Methods for Testing and Specification (MTS);The Testing and Test Control Notation version 3;Part 2: TTCN-3 Tabular presentation Format (TFT)
ES 201 873-3	Title: Methods for Testing and Specification (MTS);The Testing and Test Control Notation version 3;Part 3: TTCN-3 Graphical presentation Format (GFT)
ES 201 873-4	Title: Methods for Testing and Specification (MTS);The Testing and Test Control Notation version 3;Part 4: TTCN-3 Operational Semantics
ES 201 873-5	Title: Methods for Testing and Specification (MTS);The Testing and Test Control Notation version 3;Part 5: TTCN-3 Runtime Interface (TRI)
ES 201 873-6	Title: Methods for Testing and Specification (MTS);The Testing and Test Control Notation version 3;Part 6: TTCN-3 Control Interface (TCI)
ES 201 873-7	Title: Methods for Testing and Specification (MTS);The Testing and Test Control Notation version 3;Part 7: Using ASN.1 with TTCN-3
ES 201 873-8	Title: Methods for Testing and Specification (MTS);The Testing and Test Control Notation version 3;Part 8: The IDL to TTCN-3 Mapping
ES 201 873-9	Title: Methods for Testing and Specification (MTS);The Testing and Test Control Notation version 3;Part 9: Using XML schema with TTCN-3
ES 201 873-10	Title: Methods for Testing and Specification (MTS);The Testing and Test Control Notation version 3;Part 10: TTCN-3 Documentation Comment Specification
ES 202 553	Title: Methods for Testing and Specification (MTS);TPLan: A notation for expressing Test Purposes

SPICUM servicio de publicaciones

Código	Título
ETS 300 406	Title: Methods for Testing and Specification (MTS);Protocol and profile conformance testing specifications;Standardization methodology
ETS 300 414[2]	Title: Methods for Testing and Specification (MTS);Use of SDL in European Telecommunication Standards;Rules for testability and facilitating validation
TS 101 804-1	Title: Methods for Testing and Specification (MTS);Conformance Test Specification for ITU-T H.225.0 (Terminal, Gatekeeper and Gateway);Part 1: Protocol Implementation Conformance Statement (PICS) proforma
TS 101 804-2	Title: Methods for Testing and Specification (MTS);Conformance Test Specification for ITU-T H.225.0 (Terminal, Gatekeeper and Gateway);Part 2: Test Suite Structure and Test Purposes (TSS&TP)
TS 101 804-3	Title: Methods for Testing and Specification (MTS);Conformance Test Specification for ITU-T H.225.0 (Terminal, Gatekeeper and Gateway);Part 3: Abstract Test Suite (ATS) and partial Protocol Implementation eXtra Information for Testing (PIXIT) proforma
TS 101 875	Title: Methods for Testing and Specification (MTS);The Tree and Tabular Combined Notation version 3 TTCN-3: Library of Additional Predefined Functions
TS 101 969[2]	Title: Methods for Testing and Specification (MTS);Abstract Syntax Notation One (ASN.1) encoding rules;Specification of Encoding Control Notation (ECN) [Draft ITU-T Recommendation X.692]
TS 102 027-1	Title: Methods for Testing and Specification (MTS);Conformance Test Specification for SIP (IETF RFC 3261);Part 1: Protocol Implementation Conformance Statement (PICS) proforma
TS 102 027-2	Title: Methods for Testing and Specification (MTS);Conformance Test Specification for SIP (IETF RFC 3261);Part 2: Test Suite Structure and Test Purposes (TSS&TP)
TS 102 027-3	Title: Methods for Testing and Specification (MTS);Conformance Test Specification for SIP (IETF RFC 3261);Part 3: Abstract Test Suite (ATS) and partial Protocol Implementation eXtra Information for Testing (PIXIT) proforma
TS 102 219	Title: Methods for Testing and Specification (MTS);The IDL to TTCN-3 Mapping
TS 102 351	Title: Methods for Testing and Specification (MTS);IP Testing;TTCN-3 IPv6 Test Specification Toolkit
TS 102 369	Title: Methods for Testing and Specification (MTS);Stream Control Transmission Protocol (SCTP);Test Suite Structure and Test Purposes (TSS&TP)
TS 102 374-1	Title: Methods for Testing and Specification (MTS);Conformance Test Specification for ITU-T H.248.1;Part 1: Protocol Implementation Conformance Statement (PICS) proforma
TS 102 374-2	Title: Methods for Testing and Specification (MTS);Conformance Test Specification for ITU-T H.248.1;Part 2: Test Suite Structure and Test Purposes (TSS&TP)

[2] *Withdrawn*

SPICUM servicio de publicaciones

Código	Título
TS 102 374-3	Title: Methods for Testing and Specification (MTS);Conformance Test Specification for ITU-T H.248.1;Part 3: Abstract Test Suite (ATS) and partial Protocol Implementation eXtra Information for Testing (PIXIT) proforma
TS 102 380	Title: Methods for Testing and Specification (MTS);SS7 Message Transfer Part 2 - User Adaptation Layer (M2UA);Test Suite Structure and Test Purposes (TSS&TP)
TS 102 381	Title: Methods for Testing and Specification (MTS);SS7 Message Transfer Part 3 - User Adaptation Layer (IETF RFC 3332);Test Suite Structure and Test Purposes (TSS&TP)
TS 102 514	Title: Methods for Testing and Specification (MTS);Internet Protocol Testing (IPT): IPv6 Core Protocol;Requirements Catalogue
TS 102 515	Title: Methods for Testing and Specification (MTS);Internet Protocol Testing (IPT): IPv6 Core Protocol;Conformance Test Suite Structure and Test Purposes (TSS&TP)
TS 102 516	Title: Methods for Testing and Specification (MTS);Internet Protocol Testing (IPT): IPv6 Core Protocol;Conformance Abstract Test Suite (ATS) and partial Protocol Implementation eXtra Information for Testing (PIXIT) proforma
TS 102 517	Title: Methods for Testing and Specification (MTS);Internet Protocol Testing (IPT): IPv6 Core Protocol;Interoperability Test Suite (ITS)
TS 102 518	Title: Methods for Testing and Specification (MTS);Internet Protocol Testing (IPT);IPv4 to IPV6 Transitioning;Conformance Test Suite Structure and Test Purposes (TSS&TP)
TS 102 558	Title: Methods for Testing and Specification (MTS);Internet Protocol Testing (IPT): IPv6 Security;Requirements Catalogue
TS 102 559	Title: Methods for Testing and Specification (MTS);Internet Protocol Testing (IPT): IPv6 Mobility;Requirements Catalogue
TS 102 593	Title: Methods for Testing and Specification (MTS);Internet Protocol Testing (IPT);IPv6 Security;Conformance Test Suite Structure and Test Purposes (TSS&TP)
TS 102 594	Title: Methods for Testing and Specification (MTS);Internet Protocol Testing (IPT): IPv6 Security;Conformance Abstract Test Suite (ATS) and partial Protocol Implementation eXtra Information for Testing (PIXIT) proforma
TS 102 595	Title: Methods for Testing and Specification (MTS);Internet Protocol Testing (IPT);IPv6 Mobility;Conformance Test Suite Structure and Test Purposes (TSS&TP)
TS 102 596	Title: Methods for Testing and Specification (MTS);Internet Protocol Testing (IPT): IPv6 Mobility;Conformance Abstract Test Suite (ATS) and partial Protocol Implementation eXtra Information for Testing (PIXIT) proforma
TS 102 597	Title: Methods for Testing and Specification (MTS);Internet Protocol Testing (IPT): IPv6 Security;Interoperability Test Suite
TS 102 598	Title: Methods for Testing and Specification (MTS);Internet Protocol Testing (IPT): IPv6 Mobility;Interoperability Test Suite

SPICUM servicio de publicaciones

Código	Título
TS 102 599	Title: Methods for Testing and Specification (MTS);Internet Protocol Testing (IPT): IPv4 to IPV6 Transitioning;Requirements Catalogue
TS 102 620	Title: Methods for Testing and Specification (MTS);Internet Protocol Testing (IPT): IPv4 to IPv6 Transitioning;Interoperability Test Suite
TS 102 751	Title: Methods for Testing and Specification (MTS);Internet Protocol Testing (IPT): IPv4 to IPv6 Transitioning;Conformance Abstract Test Suite (ATS) and partial Protocol Implementation eXtra Information for Testing (PIXIT) proforma

Tabla B.3: Lista de informes y directivas producidos por TC MTS.

Código	Título
EG 201 014	Title: Methods for Testing and Specification (MTS);ETSI Standards-making;Technical quality criteria for telecommunications standards
EG 201 015	Title: Methods for Testing and Specification (MTS);Specification of protocols and services;Validation methodology for standards using Specification and Description Language (SDL);Handbook
EG 201 058	Title: Methods for Testing and Specification (MTS);Implementation Conformance Statement (ICS) proforma style guide
EG 201 148[2]	Title: Methods for Testing and Specification (MTS);Guide for the use of the second edition of TTCN
EG 201 383	Title: Methods for Testing and Specification (MTS);Use of SDL in ETSI deliverables;Guidelines for facilitating validation and the development of conformance tests
EG 201 872	Title: Methods for Testing and Specification (MTS);Methodological approach to the use of object-orientation in the standards making process
EG 202 103	Title: Methods for Testing and Specification (MTS);Guide for the use of the second edition of TTCN
EG 202 106	Title: Methods for Testing and Specification (MTS);Guidelines for the use of formal SDL as a descriptive tool
EG 202 107	Title: Methods for Testing and Specification (MTS);Planning for validation and testing in the standards-making process
EG 202 237	Title: Methods for Testing and Specification (MTS);Internet Protocol Testing (IPT);Generic approach to interoperability testing
EG 202 568	Title: Methods for Testing and Specification (MTS);Internet Protocol Testing (IPT);Testing: Methodology and Framework
ETR 071	Title: Methods for Testing and Specification (MTS);Semantic relationship between SDL and TTCN;A common semantics representation
ETR 094	Title: Methods for Testing and Specification (MTS);Guide for the implementation of ISO/IEC 9646 Conformance Assessment Process (CAP)

SPICUM servicio de publicaciones

Código	Título
ETR 130	Title: Methods for Testing and Specification (MTS);Interoperability and conformance testing A classification scheme
ETR 141	Title: Methods for Testing and Specification (MTS);Protocol and profile conformance testing specifications;The Tree and Tabular Combined Notation (TTCN) style guide
ETR 142	Title: Methods for Testing and Specification (MTS);Guidance on the production and completion of System Conformance Statement (SCS) proforma
ETR 153	Title: Methods for Testing and Specification (MTS);Guidance on the production and completion of System Conformance Test Report (SCTR) and Protocol Conformance Test Report (PCTR) proforma
ETR 171	Title: Methods for Testing and Specification (MTS);Proforma for scoping reports
ETR 184	Title: Methods for Testing and Specification (MTS);Overview of validation techniques for European Telecommunication Standards (ETSs) containing SDL
ETR 190	Title: Methods for Testing and Specification (MTS);Partial and multi-part Abstract Test Suites (ATS);Rules for the context-dependent re-use of ATSs
ETR 193	Title: Methods for Testing and Specification (MTS);Network Integration Testing (NIT);Methodology aspects;Test Co-ordination Procedure (TCP) style guide
ETR 212[3]	Title: Methods for Testing and Specification (MTS);Implementation Conformance Statement (ICS) proforma style guide
ETR 259	Title: Methods for Testing and Specification (MTS);Bibliography of techniques, methodologies and practices used in Standards making
ETR 266	Title: Methods for Testing and Specification (MTS);Test Purpose style guide
ETR 267	Title: Methods for Testing and Specification (MTS);Conformance testing specifications;Quality review tables
ETR 298	Title: Methods for Testing and Specification (MTS);Specification of protocols and services;Handbook for SDL, ASN.1 and MSC development
ETR 303	Title: Methods for Testing and Specification (MTS);Test Synchronization;Architectural reference;Test Synchronization Protocol 1 (TSP1) specification
ETR 304	Title: Methods for Testing and Specification (MTS);The future in ETSI of quality of standards-making, validation and testing
TCRTR 006	Title: Methods for Testing and Specification (MTS);ETSI and certification in telecommunications;Overview of outstanding issues and some recommendations
TCRTR 017	Title: Methods for Testing and Specification (MTS);Technical Basis for Regulation (TBR) specification methodology
TCRTR 048	Title: Methods for Testing and Specification (MTS);Implementation Conformance Statement (ICS) proforma style guide [endorsement of ETR 212 (1995)]

[3] *Historical*

SPICUM servicio de publicaciones

Código	Título
TCRTR 050	Title: Methods for Testing and Specification (MTS);Specification of protocols and services;Handbook for SDL, ASN.1 and MSC development [ETR 298 (1996)]
TCRTR 052	Title: Methods for Testing and Specification (MTS);Conformance testing specification;Quality review tables [ETR 267 (1996)]
TR 101 023-1	Title: Methods for Testing and Specification (MTS);Portability of SDL specifications
TR 101 023-2	Title: Methods for Testing and Specification (MTS);Application of object-oriented SDL features in B-ISDN specifications
TR 101 028	Title: Methods for Testing and Specification (MTS);Survey on the Use of Test Specifications produced by ETSI
TR 101 034	Title: Methods for Testing and Specification (MTS);Reports on experiments in validation methodology;PISN Cordless Terminal Mobility Incoming Call Additional Network Feature (ANF-CTMI)
TR 101 051	Title: Methods for Testing and Specification (MTS);Report of the CATG[4] applications
TR 101 081	Title: Methods for Testing and Specification (MTS);Strategy for the use of formal SDL for descriptive purposes in ETSI products
TR 101 101	Title: Methods for Testing and Specification (MTS);TTCN interim version including ASN.1 1994 support [ISO/IEC 9646-3] (Second Edition Mock-up for JTC1/SC21 Review)
TR 101 114	Title: Methods for Testing and Specification (MTS);Analysis of the use of ASN.1 94 with TTCN and SDL in ETSI deliverables
TR 101 295	Title: Methods for Testing and Specification (MTS);Rules for the transportation of ASN.1 definitions using X.681, X.682 and X.683 to equivalent X.680 constructs
TR 101 667	Title: Methods for Testing and Specification (MTS);Network Integration Testing (NIT);Interconnection;Reasons and goals for a global service testing approach
TR 101 680	Title: Methods for Testing and Specification (MTS);A harmonized integration of ASN.1, TTCN and SDL
TR 101 778	Title: Methods for Testing and Specification (MTS);Technical quality of telecommunications standards;Complementary methods and technical quality manual
TR 101 873-3	Title: Methods for Testing and Specification (MTS);The Testing and Test Control Notation version 3;Part 3: TTCN-3 Graphical presentation Format (GFT)
TR 101 874	Title: Methods for Testing and Specification (MTS);The Tree and Tabular Combined Notation version 3;TTCN-3: TTCN-2 to TTCN-3 Mapping
TR 102 043	Title: Methods for Testing and Specification (MTS);The TTCN-3 Runtime Interface (TRI);Concepts and definition of the TRI

[4] CATG - *Computer Aided Test Generator*.

SPICUM servicio de publicaciones

Código	Título
TR 102 105	Title: Methods for Testing and Specification (MTS);Methodological approach to the use of object-orientation within the standards making process;Initial study
TR 102 200	Title: Methods for Testing and Specification (MTS);UMTS Testing Methodology
TR 102 205	Title: Methods for Testing and Specification (MTS);UML 2.0 action syntax feasibility study
TR 102 235	Title: Methods for Testing and Specification (MTS);Internet Protocol Testing (IPT);Pre-normative Study for IPv6 Testing
TR 102 422	Title: Methods for Testing and Specification (MTS) IMS Network Integration Testing Infrastructure Testing Methodology
TR 102 560	Title: Methods for Testing and Specification (MTS);URI name space for TTCN-3
TR 102 743	Title: Methods for Testing and Specification (MTS);Recommendations for improvements to the ETSI Standards Engineering Process
TR 121 900	Title: Digital cellular telecommunications system (Phase 2+);Universal Mobile Telecommunications System (UMTS);Technical Specification Group working methods (3GPP TR 21.900 version 3.7.0 Release 1999)
TR 121 900	Title: Digital cellular telecommunications system (Phase 2+);Universal Mobile Telecommunications System (UMTS);Technical Specification Group working methods (3GPP TR 21.900 version 4.1.0 Release 4)
TR 121 900	Title: Digital cellular telecommunications system (Phase 2+);Universal Mobile Telecommunications System (UMTS);Technical Specification Group working methods (3GPP TR 21.900 version 5.1.0 Release 5)
TR 121 900	Title: Digital cellular telecommunications system (Phase 2+);Universal Mobile Telecommunications System (UMTS);Technical Specification Group working methods (3GPP TR 21.900 version 6.4.0 Release 6)
TR 121 900	Title: Digital cellular telecommunications system (Phase 2+);Universal Mobile Telecommunications System (UMTS);Technical Specification Group working methods (3GPP TR 21.900 version 7.3.0 Release 7)
TR 121 900	Title: Digital cellular telecommunications system (Phase 2+);Universal Mobile Telecommunications System (UMTS);Technical Specification Group working methods (3GPP TR 21.900 version 8.2.0 Release 8)

SPICUM servicio de publicaciones

Apéndice

C

APÉNDICE C: MÉTODOS DE PRUEBAS DE SISTEMAS DE COMUNICACIONES INALÁMBRICAS

En este Apéndice se presentan los Métodos de Pruebas utilizados en la certificación de equipos de los sistemas de comunicaciones sobre los que se ha trabajado en esta Tesis. Se ha incluido también el sistema GSM para completar la familia europea de sistemas inalámbricos: DECT, GSM, UMTS. El último apartado se dedica a la tecnología Bluetooth. Como se puede observar, el Método de Pruebas remoto es la configuración más utilizada.

C.1 DECT

ETSI ha definido Juegos de Pruebas tanto para la Terminación Fija como la Terminación Portátil dado que ambos equipos pueden ser adquiridos por el usuario en una tienda. Existen 5 Juegos de Pruebas para los niveles de protocolo del sistema DECT (Figura C.1), ya que el Nivel DLC emplea el mismo para ambas Terminaciones. Los correspondientes Métodos de Pruebas Abstractas se muestran en la Figura C.2, la Figura C.3 y la Figura C.4. Todos utilizan el Método de Pruebas remoto en la variante empotrada.

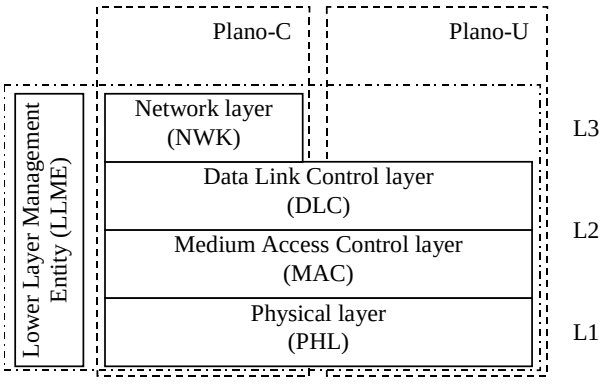


Figura C.1: Arquitectura del sistema DECT [ETS 300 175-1].

SPICUM servicio de publicaciones

Los Juegos de Pruebas son los siguientes:

- Nivel MAC
  - Terminación Portátil: [EN 300 497-2]
  - Terminación Fija: [EN 300 497-3]
- Nivel DLC
  - Terminaciones Portátil y Fija: [EN 300 497-5]
- Nivel NWK
  - Terminación Portátil: [EN 300 497-7]
  - Terminación Fija: [EN 300 497-9]

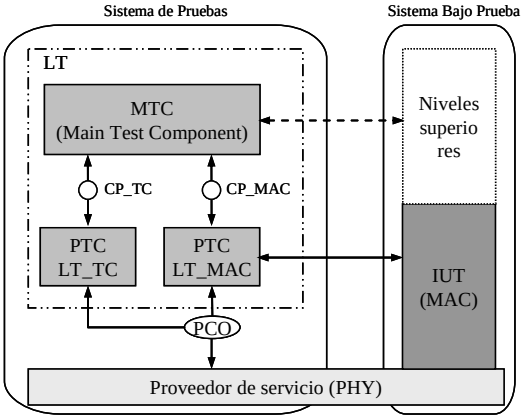


Figura C.2: Método de Pruebas para el Nivel MAC (FT/PT).

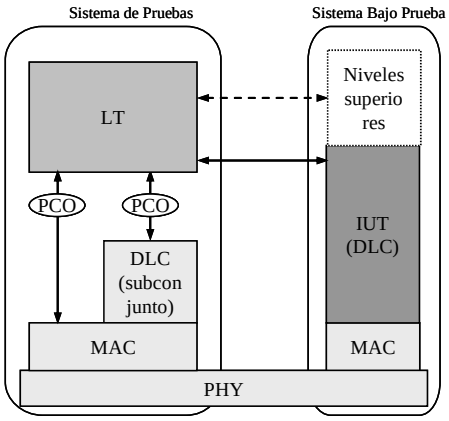


Figura C.3: Método de Pruebas para el Nivel DLC (FT/PT).

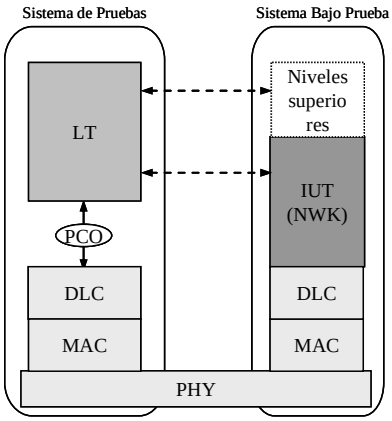


Figura C.4: Método de Pruebas para el Nivel NWK (FT/PT).

C.2 GSM

El diseño del sistema GSM (*Global System for Mobile communications*) coincidió con la estandarización de la Metodología OSI de Pruebas de Conformidad, por lo que ha

SPICUM servicio de publicaciones

sido de los primeros sistemas en incorporar dicha metodología en su fase de certificación de productos.

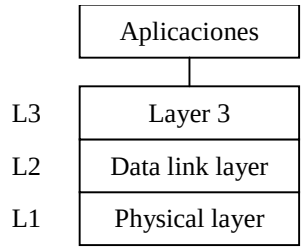


Figura C.5: Arquitectura de una estación móvil GSM [ETS 300 550].

La arquitectura de un móvil (MS – *Mobile Station*) GSM se muestra en la Figura C.5. Las pruebas de conformidad para los niveles de protocolo se encuentran definidas en [EN 300 607-1]. Para el Nivel L2 se especifican, para cada Caso de Prueba, el procedimiento de prueba y la secuencia de tramas L2 que el Sistema de Pruebas debe generar y recibir. No existe un Juego de Pruebas en TTCN para el Nivel L2. Para el Nivel L3, el Juego de Pruebas está especificado en [EN 300 607-3]. El Nivel L2 emplea el Método de Pruebas remoto, mientras que el Nivel L3 emplea el Método de Pruebas distribuido (Figura C.6).

Se puede observar en la Figura C.6 que el Sistema de Pruebas, denominado también Simulador del Sistema (SS – *System Simulator*), se comunica con los niveles inferiores haciendo uso de un doble mecanismo. A través del PCO se transmite y recibe la señalización habitual de la capa superior, estableciendo la comunicación con el Sistema Bajo Prueba. Mediante el bloque `Gestión` los Juegos de Pruebas pueden modificar el comportamiento del Subsistema Inferior. La interfaz que ofrece este bloque es una interfaz definida en los Juegos de Pruebas mediante funciones TSO, no mediante primitivas.

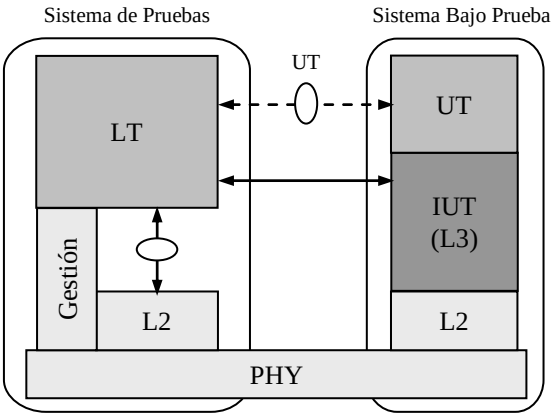


Figura C.6: Método de prueba para el Nivel L3 de GSM.

SPICUM servicio de publicaciones

C.3 UMTS

La arquitectura de la interfaz radio de un terminal UMTS es la mostrada en la Figura C.7. En UMTS se ha especificado un Juego de Pruebas para cada uno de los niveles de protocolo de su arquitectura. La filosofía que ha regido este diseño ha sido disponer de una única arquitectura de pruebas, de forma que los puntos de observación y control entre cada Juego de Pruebas y el Proveedor de Servicio sean los mismos en todas circunstancias. Esta arquitectura se puede observar en la Figura C.8.

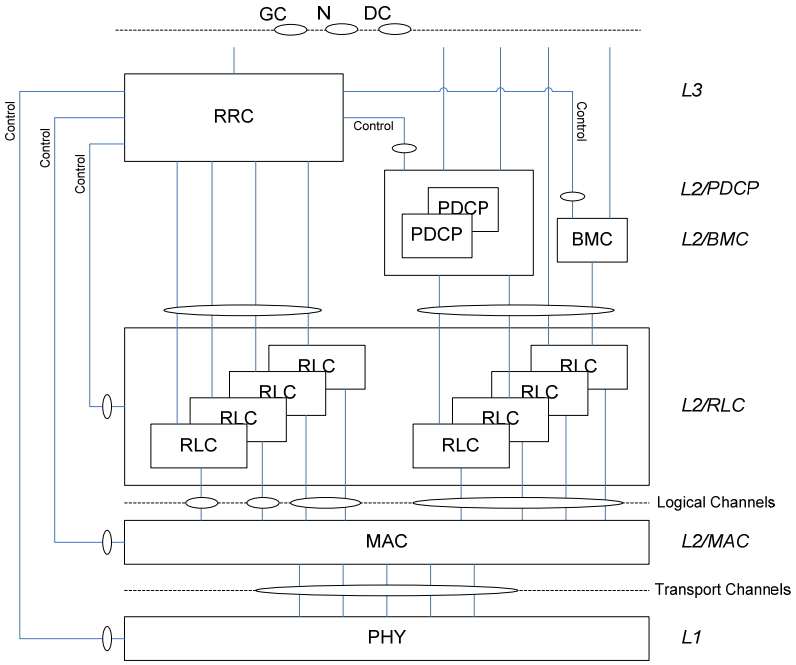


Figura C.7: Arquitectura de la interfaz radio de UMTS.

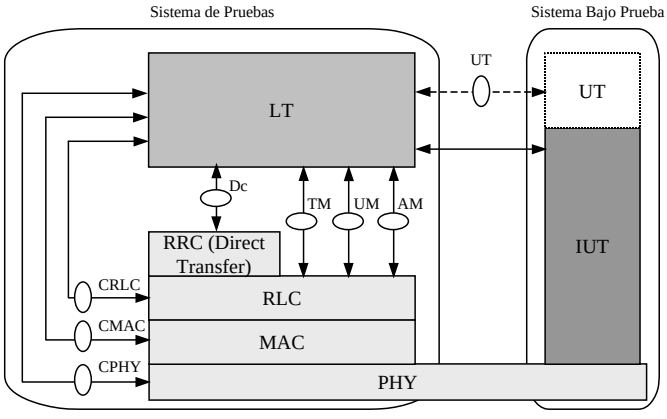


Figura C.8: Arquitectura única de pruebas para UMTS.

Los Métodos de Pruebas se describen en [3GPP 34.123-3]. Para los protocolos del Nivel 2 se utiliza el Método de Pruebas remoto: MAC (Figura C.9), RLC (Figura C.10), PDCP (Figura C.11) y BMC (Figura C.12). Para el Nivel 3 y superiores se emplea el Método de Pruebas distribuido: RRC (Figura C.13), NAS y SMS (Figura C.14).

SPICUM servicio de publicaciones

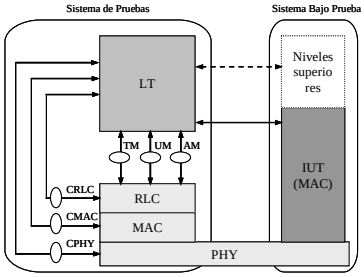


Figura C.9: Método de Pruebas para el Nivel MAC.

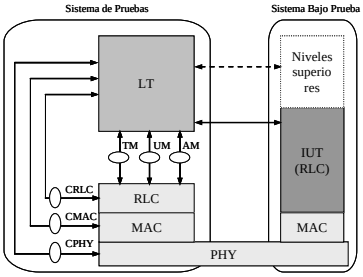


Figura C.10: Método de Pruebas para el Nivel RLC.

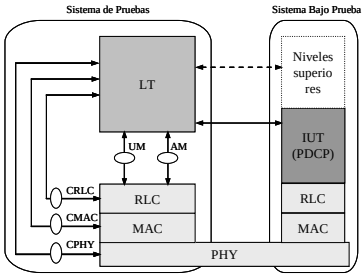


Figura C.11: Método de Pruebas para el Nivel PDCP.

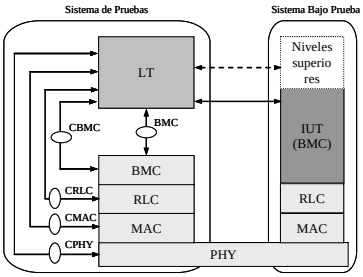


Figura C.12: Método de Pruebas para el Nivel BMC.

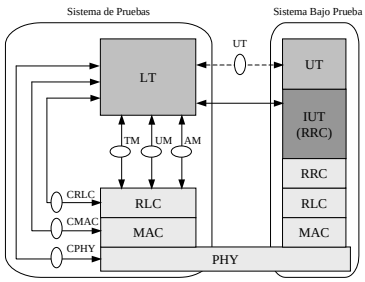


Figura C.13: Método de Pruebas para el Nivel RRC.

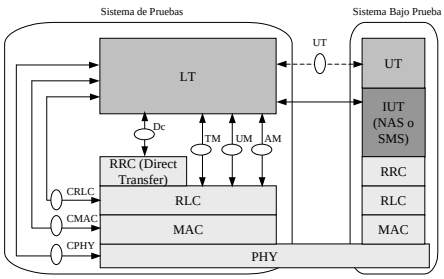


Figura C.14: Método de Pruebas para las funcionalidades NAS y SMS.

C.4 Bluetooth

La arquitectura del sistema Bluetooth es la mostrada en la Figura C.15. El SIG (*Bluetooth Special Interest Group*), como organismo responsable del desarrollo de la tecnología, ha definido Juegos de Pruebas[1] de conformidad para los niveles y perfiles Banda Base [BTEST BB], LM [BTEST LM], L2CAP [BTEST L2CAP], SDP [BTEST SDP], GAP [BTEST GAP] y SPP [BTEST SPP]. Los correspondientes Métodos de Pruebas Abstractas se muestran de la Figura C.16 a la Figura C.21. Se utiliza la variante

[1] Cualquier dispositivo Bluetooth puede actuar en el papel de maestro o de esclavo, por lo que los Juegos de Pruebas contienen Casos de Prueba específicos para cada configuración.

SPICUM servicio de publicaciones

empotrada en configuración local (BB, L2CAP, LM) o en configuración remota (SDP, GAP, SPP).

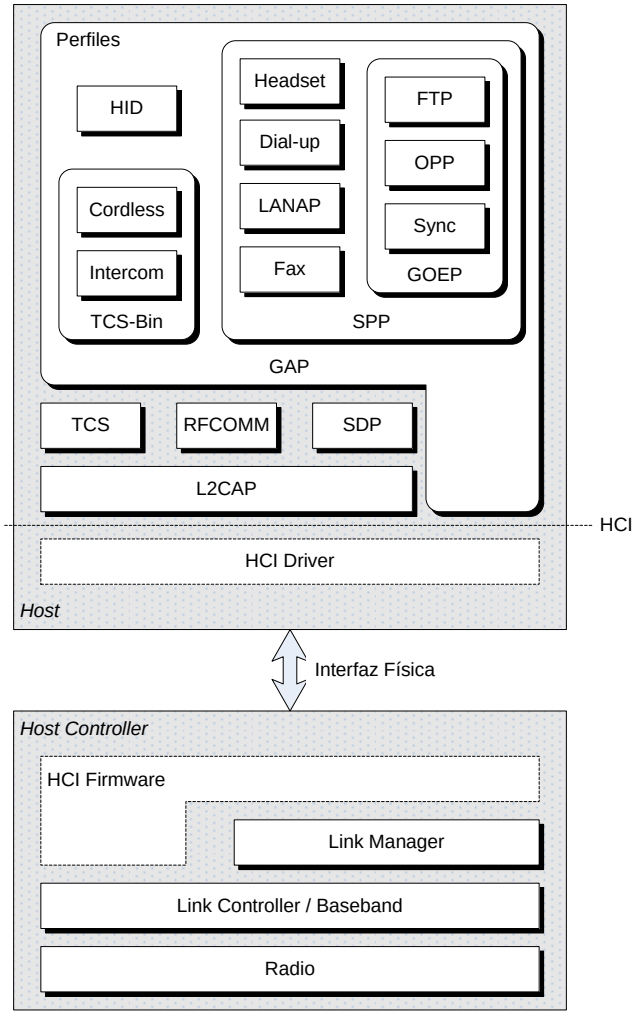


Figura C.15: Arquitectura del sistema Bluetooth.

SPICUM servicio de publicaciones

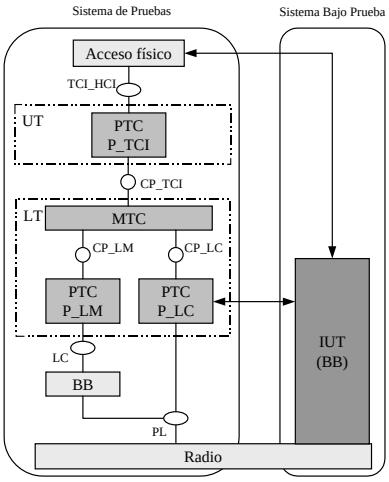


Figura C.16: Método de Pruebas para el Nivel Banda Base.

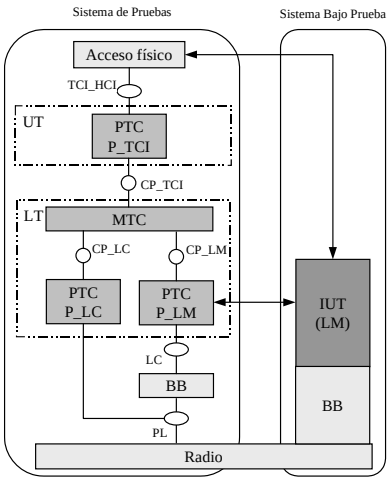


Figura C.17: Método de Pruebas para el Nivel LM.

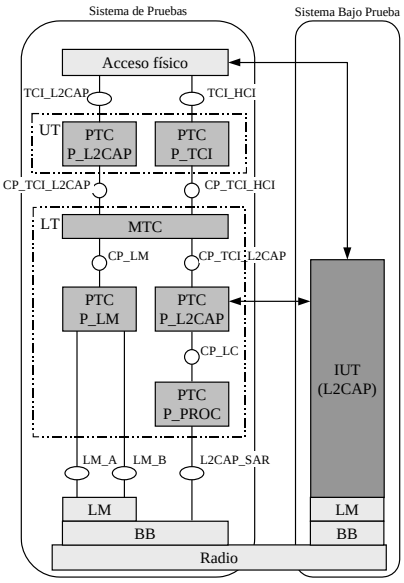
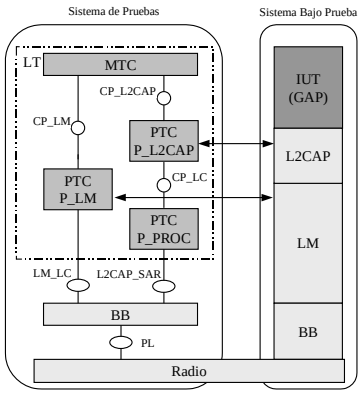
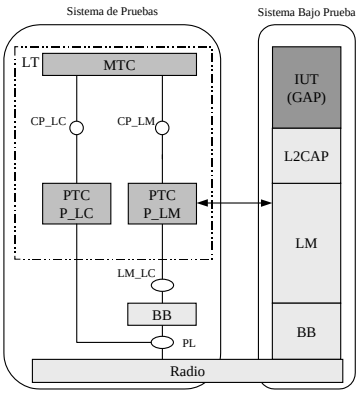


Figura C.18: Método de Pruebas para el Nivel L2CAP.





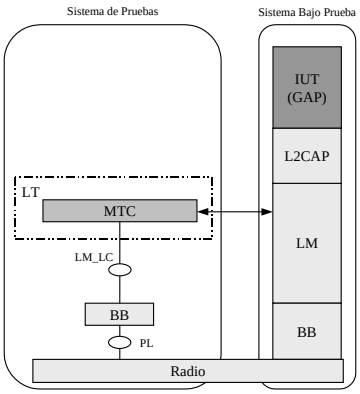


Figura C.19: Métodos de Pruebas para el perfil GAP en configuraciones MTC_ L2CAP_PLM_CONFIG, MTC_PLM_PLC_CONFIG y no concurrente.

SPICUM servicio de publicaciones

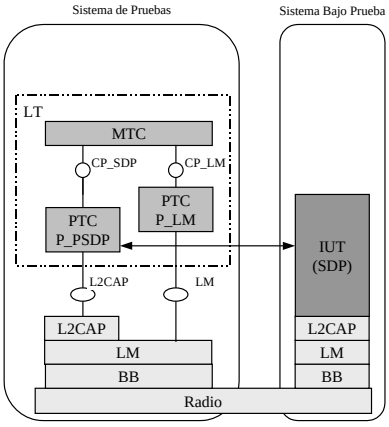


Figura C.20: Método de Pruebas para el Nivel SDP.

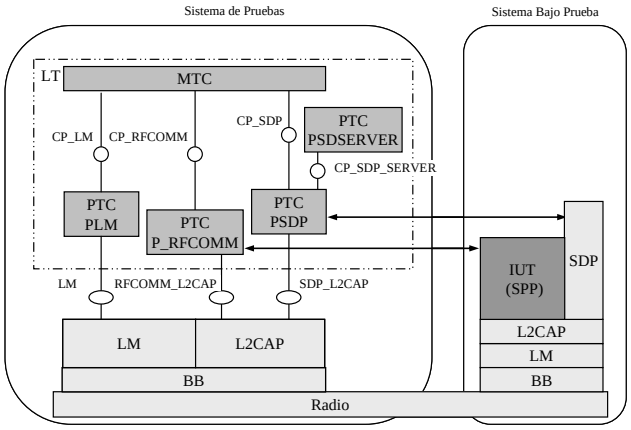


Figura C.21: Método de Pruebas para el Nivel SPP.

SPICUM servicio de publicaciones

Apéndice

D

APÉNDICE D: FUNCIONES DE LA INTERFAZ GCI

La Interfaz GCI [GCI96] para TTCN versión 2 incluye las funciones indicadas en las siguientes tablas. Al final del Apéndice se listan las funciones que hay que implementar en el Módulo Adaptador de las Pruebas si se utiliza el generador de código de la herramienta Tau.

Tabla D.1: Interfaz de Operación de la Interfaz GCI.

Función GCI	Descripción
`GciReadTSPar`	Lectura del valor de un Parámetro de Prueba.
`GciSnapshot`	Recoge los eventos que hay en las colas de entrada o de vencimiento de temporizadores al ser llamada.
`GciSend`	Envío de un dato hacia un PCO o CP concreto.
`GciStartTimer`	Arranque de un temporizador.
`GciCancelTimer`	Cancelación de un temporizador.
`GciReadTimer`	Lectura del valor de un temporizador.
`GciLog`	Envío de un mensaje de traza.
`<Funcion_TSO>`	Una función para cada TSO del Juego de Pruebas.
`GciConfigure`	Indica la configuración de Componentes de Prueba que un Caso de Prueba va a utilizar. Se invoca antes de iniciar la ejecución.
`GciCreate`	Crea un Componente de Prueba.
`GciTestComponentDone`	Indica al Módulo Adaptador de las Pruebas que un Componente de Prueba ha finalizado; devuelve el resultado de su ejecución.

Tabla D.2: Interfaz de Comportamiento de la Interfaz GCI.

Función GCI	Descripción
`GciReceive`	Recepción en el TTCN-RB de un valor procedente de un determinado PCO o CP.
`GciTimeout`	Comprueba si ha vencido un temporizador.
`GciDone`	El Módulo Adaptador de las Pruebas informa al módulo TTCN-RB de que un Componente de Prueba ha finalizado.

SPICUM servicio de publicaciones

Tabla D.3: Interfaz de Gestión de la Interfaz GCI.

	Función GCI	Descripción
	GciInit	Inicialización del TTCN-RB.
	GciStartTestCase	Ejecución de un Caso/Grupo de Pruebas.
	GciGetTestCaseList	Obtiene la lista de Casos de Prueba disponibles.
	GciGetTestCaseGroupList	Obtiene la lista de Grupos de Pruebas.
	GciGetNoOfTimers	Devuelve el número de temporizadores.
	GciGetTimerName	Obtiene el nombre asociado a un temporizador.
	GciGetNoOfPCOs	Devuelve el número de PCOs.
	GciGetPCOName	Obtiene el nombre asociado a un PCO.
	GciGetPCOType	Obtiene el tipo de un PCO.
	GciGetPCORole	Obtiene el papel (LT o UT) que juega un PCO.
	GciGetValue	Obtiene el valor asociado de una variable TTCN.
TTCN Concurrente	GciStartTestComponent	Inicia la ejecución de un Componente de Prueba.
	GciGetNoOfComponents	Obtiene el número de Componentes de Prueba.
	GciGetComponent	Devuelve un puntero a un Componente de Prueba.
	GciGetComponentName	Obtiene el nombre del Componente de Prueba.
	GciGetComponentIdentifier	Devuelve el identificador del Componente de Prueba.
	GciGetComponentType	Devuelve el tipo del Componente de Prueba.
	GciGetComponentNoOfCPs	Obtiene el número de CPs de un Componente de Prueba.
	GciGetComponentCP	Devuelve un CP de un Componente de Prueba.
	GciGetComponentNoOfPCOs	Obtiene el número de PCOs de un Componente de Prueba.
	GciGetComponentPCO	Devuelve un PCO de un Componente de Prueba.
Obsoletas	GciGetResultVerdict	Obtiene el veredicto de una Prueba.
	GciIsResultError	Indica si el resultado de una Prueba ha sido erróneo.
	GciGetResultMessage	Obtiene el mensaje asociado al resultado de una Prueba.
	GciGetResultValue	Obtiene una referencia a un resultado.
	GciGetTestListNoOfNames	Devuelve el número de Casos de Prueba que hay en una lista.
	GciGetTestListName	Devuelve el nombre del Caso de Prueba contenido en una posición dada de una lista.

Tabla D.4: Interfaz de Valores de la Interfaz GCI.

Función GCI	Descripción
GciMk<tipo>	Crea un valor de tipo <tipo>.
GciGet<tipo>	Lee el contenido de un valor de tipo <tipo>.
GciGetType	Obtiene el tipo de un valor.
GciSetType	Fija el tipo de un valor.
GciTagType	Fija la etiqueta ASN.1 de un valor.
GciGetTag	Obtiene el nombre del tipo de un valor.
GciGetClass	Obtiene la etiqueta ASN.1 de un valor.
GciGetImplicit	Indica si el tipo es implícito o no[1].
GciMkEXTERNAL	Convierte una variable del Módulo Adaptador de las Pruebas en un valor utilizable por el módulo TTCN-RB.
GciGetEXTERNAL	Convierte un valor del módulo TTCN-RB en una variable utilizable por el Módulo Adaptador de las Pruebas.
GciSetEncoding	Fija la codificación a usar para un valor.
GciGetEncoding	Obtiene la codificación a usar para un valor.
GciMkPreEncoded	Crea un valor cuyo contenido ya está codificado.
GciIsPreEncoded	Indica si el valor ya está codificado.
GciGetSymbolicIdentifier	Obtiene información adicional sobre la codificación de un valor.

SPICUM servicio de publicaciones

Tabla D.5: Interfaz de Valores, para tipos de datos estructurados, de la Interfaz GCI.

	Función GCI	Descripción
	Interfaz flexible	
SEQUENCE	GciMkSEQUENCE	Crea un valor de tipo SEQUENCE.
	GciSetField	Asigna el valor de un campo.
	GciUnsetField	Elimina el campo indicado.
	GciGetField	Lee el valor de un campo.
	GciSeqSize	Devuelve el número de campos en el tipo.
	GciIsPresent	Indica si un campo está presente.
	GciGetFieldByIndex	Devuelve el valor del campo que aparece en la posición indicada.[2]
SEQUENCE_OF	GciMkSEQUENCEOF	Crea un valor de tipo SEQUENCE OF.
	GciAddElem	Añade el valor al final de la secuencia.
	GciGetElem	Lee el valor de un elemento de la secuencia.
	GciSeqOfSize	Obtiene el número de elementos presentes.
	Interfaz estricto	
	GciMk_<tipo>	Crea un valor del tipo <tipo> indicado.
	GciSet_<tipo>_<campo>	Asigna un valor al campo <campo> de un valor del tipo <tipo>.
	GciGet_<tipo>_<campo>	Lee el valor del campo <campo> de un valor del tipo <tipo>.
	GciAdd_<tipo>_Elem	Añade un nuevo elemento a la secuencia de tipo <tipo>.
	GciGet_<tipo>_Elem	Lee un elemento de un valor de tipo <tipo>.
	GciSize_<tipo>	Obtiene el número de elementos presentes en un valor de tipo <tipo>.

D.1 Módulo Adaptador de las Pruebas

Las siguientes tablas listan las funciones de la Interfaz GCI que hay que implementar en el Módulo Adaptador de las Pruebas si se utiliza la herramienta Tau para generar código a partir de los Juegos de Pruebas Abstractas escritos en TTCN.

Tabla D.6: Funciones en el Módulo de Gestión (TAM – Test Adaptor Management).

AdClearActStep	GciConfig	Taskcleanup
AdExit	GciCreate	Taskcreate
AdInit	GetCompActStepByThreadId	Taskdelete
AdInitThreads	GetCompNameByThreadId	Taskinit
		Taskreschedule

Tabla D.7: Funciones en el Módulo de Codificación y Descodificación (TAC – Test Adaptor Codec).

AdClosePCOChannels	AdInitTimers	GciCancelTimer
AdCreateTimer	AdReceiveASP	GciImplicitSend
AdGetElapsedTime	AdRemoveAllPCOs	GciReadTimer
AdGetPCO	AdRemoveAllTimers	GciSend
AdGetPCOReadFds	AdUpdatePCOs	GciSnapshot
AdGetTimer	AdUpdateTimers	GciStartTimer

SPICUM servicio de publicaciones

Tabla D.8: Módulo de Registro (TAL – Test Adaptor Logging).

GciLog	GciLogPrint	GciReadTSPar

SPICUM servicio de publicaciones

Apéndice

E

APÉNDICE E: IMPLEMENTACIÓN DE GENERADORES DE CODIFICADORES

En este Apéndice se describen posibles alternativas para implementar la herramienta *GenDef* (Capítulo 5, Sección 5.5.2) y la solución empleada.

E.1 Alternativas de Implementación

Una versión sencilla de la herramienta *GenDef* se puede realizar rápidamente, en un lenguaje como C, recorriendo consecutivamente cada una de las secciones del Módulo TTCN (Capítulo 5, Sección 5.5.2.1.2) anteriores y extrayendo la información que se ha resaltado. El esquema de esta versión simple sería el mostrado en la Figura E.1. Se busca en primer lugar la sección de Tipos de Datos Simples, indicada con la palabra clave de inicio `$Begin_SimpleTypeDefs`, luego se procesan todas las definiciones de Tipos Simples, hasta encontrar la palabra clave de fin de sección `$End_SimpleTypeDefs`. Después se procesan la sección de Tipos Estructurados, la de Tipos ASN.1, y así hasta terminar con la sección de definiciones de Unidades de Datos del Protocolo, en concreto, con la subsección dedicada a las definiciones por referencia en notación ASN.1.

```
for (secciones)
    Buscar (Palabra_Clave_de_Inicio_Sección)
    while Not(Palabra_Clave_de_Fin_Sección)
           for (Palabras_Clave_intermedias)
                  Extrae_Información
```

Figura E.1: Algoritmo básico de la herramienta GenDef.

Este algoritmo, sin embargo, no proporciona la generalidad que permite la gramática TTCN, ya que la mayoría de las secciones anteriores pueden repetirse un número indeterminado de veces en un fichero `*.MP`, siguiendo el formato indicado en las reglas de la Figura E.2. Estas reglas indican que dentro de cada sección pueden definirse elementos o grupos de elementos, donde cada grupo puede anidar más grupos y/o elementos y, además, puede haber tantas subsecciones como se desee y sin ningún

SPICUM servicio de publicaciones

orden. Un ejemplo donde se aplica este esquema es en los grupos de definiciones de PDUs.

```
SeccionDefs           ::=   $SeccionDefs { SeccionDefOrGroup }+
                            $End_SeccionDefs
SeccionDefOrGroup     ::=   SeccionDef | SeccionGroup
SeccionGroup          ::=   $SeccionGroup SeccionGroupId
                            {SeccionDefOrGroup}+ $End_SeccionGroup
```

Figura E.2: Esquema genérico de reglas de producción para el uso de Grupos de Elementos en TTCN.

Por tanto, se requiere una solución más genérica. Se puede pensar en una primera instancia en incluir la recursividad que hace falta en el algoritmo anterior. En este caso, la herramienta poseerá un funcionamiento[1] como el esquematizado en la Figura E.3. Cada regla corresponde a una rutina, encargada del procesamiento de las líneas de un archivo que correspondan a dicha regla. Durante el procesamiento, se mira el siguiente *token*. Si es una palabra clave, se invoca la rutina de la regla correspondiente; si no, se procesa la información, bien almacenándola, bien obviándola. Este algoritmo es recursivo pues puede ocurrir, como en las reglas mostradas en la Figura E.2, que una regla contenga reglas que, a su vez, hacen uso de la primera; en este caso, la misma rutina puede ser llamada múltiples veces sin haber retornado.

```
While Not(Fin_de_Sección)
    MiraSiguienteToken (token)
    Switch (token)
           case PalClave
                  Invocar_ProcesaSeccion_PalClave
           case Informacion
                  Extrae_Informacion
```

Figura E.3: Algoritmo recursivo de la herramienta GenDef.

Por ejemplo, la regla 73 de la gramática TTCN (Figura E.4) se correspondería con la rutina mostrada en la misma figura. Como se ve, la sintaxis de la regla gramatical se escribe en el propio flujo secuencial del programa. Esto hace que las construcciones propias de la notación EBNF deban modelarse en este flujo, como ocurre con la construcción '`{...}+`', que se ha traducido en un bucle `While`. Para que el algoritmo mostrado funcione, cada rutina que procese una regla debe terminar habiendo mirado el siguiente *token* que aparezca en el archivo.

Este esquema tiene la ventaja de ser fácilmente comprensible incluso por programadores con nula experiencia en analizadores sintácticos de lenguajes. Sin embargo, como vemos, tiene limitaciones importantes.

Para un adecuado análisis léxico, hay que definir qué es un *token*. En general, es fácil leer una palabra que corresponda a un *token*, dado que existen funciones específicas para ello, pero no es tan evidente comprender que un signo de puntuación (por ejemplo, un punto '.' o un '=') sea un *token*. Habría que escribir una función que leyera *tokens* de acuerdo con las reglas sintácticas de la notación.

[1] Una presentación más detallada de este tipo de implementaciones se encuentra en [HOLU90].

SPICUM servicio de publicaciones

Igualmente, si se encuentra un error sintáctico es interesante conocer en qué sitio se ha producido. Para ello, la herramienta debe llevar un indicador de en qué posición del fichero se encuentra, con objeto de informar al usuario.

```
(73) SimpleTypeDefs ::= $Begin_SimpleTypeDefs [SimpleTypeGroupRef]
{[CollComment]  SimpleTypeDef}+ [Comment] $End_SimpleTypeDefs

        ____________________________________________________

Procesa_SimpleTypeDefs {
    CogeToken (token) /* Es la palabra clave inicial de sección */
    MiraSiguienteToken (token)
    If (token == PalabraClave_Inicio_SimpleTypeGroupRef)
         Procesa_SimpleTypeGroupRef
    While ((token == PalabraClave_Inicio_CollComment) OR
           (token == PalabraClave_Inicio_SimpleTypeDef))
         If (token == PalabraClave_Inicio_CollComment)
              Procesa_CollComment
         If (token == PalabraClave_Inicio_SimpleTypeDef)
              Procesa_SimpleTypeDef
    If (token == PalabraClave_Inicio_Comment)
         Procesa_Comment
    CogeToken (token) /* Es la palabra clave final de sección */
    MiraSiguienteToken (token)
}
```

Figura E.4: Regla número 73 de la notación TTCN y una posible implementación.

```
/* Regla 680 */
@ Event = Send | Receive | Otherwise | Timeout | Done

/* Regla 682 */
@ Send = [PCO_Identifier | CP_Identifier | FormalParIdentifier]
@       <'!'> { ASP_Identifier | PDU_Identifier | CM_Identifier }

/* Regla 684 */
@ Receive = [PCO_Identifier | CP_Identifier | FormalParIdentifier]
@       <'?'> { ASP_Identifier | PDU_Identifier | CM_Identifier }

/* Regla 685 */
@ Otherwise = [PCO_Identifier | CP_Identifier | FormalParIdentifier]
@       <'?'> __OTHERWISE__

/* Regla 686 */
@ Timeout = <'?'> __TIMEOUT__
@      [TimerIdentifier | FormalParIdentifier]

/* Regla 687 */
@ Done = <'?'> __DONE__ <'('> [TCompIdList] <')'>
```

Figura E.5: Ejemplo de ramas ambiguas sin mirar hacia delante más de un token.

La herramienta esquematizada tampoco incluye la posibilidad de vuelta atrás. Supongamos el conjunto de reglas de la gramática TTCN mostradas en la Figura E.5. Al llegar a la regla 680, si sólo se mira un *token* hacia delante, el analizador se encuentra en

SPICUM servicio de publicaciones

una posición de indecidibilidad si el *token* leído es un interrogante '?'[2]. Cualquiera de las cuatro últimas alternativas podría ser la correcta. Lo mismo pasa si se lee un identificador, pero en este caso la indecidibilidad afectaría a las tres primeras alternativas. Hay dos posibles soluciones a este problema:

a) Reorganizar las reglas de producción: requiere analizar las reglas de la gramática y reconstruirlas de tal modo que las situaciones de indecidibilidad antes señaladas no puedan llegar a producirse.

b) Dotar al analizador de un mecanismo de vuelta atrás: el analizador debe mantener una pila de *tokens* mientras no sea capaz de decidir a qué rama corresponden. Si se encuentra con un *token* no permitido en una rama, vuelve hacia atrás hasta la última bifurcación y prueba con el resto de las ramas hasta determinar si hay alguna de ellas que corresponda a la secuencia de *tokens* leída.

Los problemas aquí planteados ocasionan que la herramienta de análisis tenga una alta complejidad, que es difícil de resolver si se parte de cero. Sin embargo, estos escollos se solucionan con el planteamiento expuesto en la siguiente Sección, que hace uso de herramientas específicas, ya existentes, para el diseño de analizadores léxicos, sintácticos y semánticos.

E.2 Generadores de Analizadores

La teoría de analizadores y compiladores ([TREM85], [AHO86]) está lo suficientemente asentada como para que no sea necesario partir de cero a la hora de construir una herramienta de análisis léxico para gramáticas similares a la gramática de TTCN. Genéricamente se utiliza el término compilador de compiladores o generador de compiladores. Son herramientas que, a partir de una gramática, generan un código que permite analizar la sintaxis y la semántica de código escrito de acuerdo a dicha gramática. El objetivo que se persigue por nuestra parte no es tan amplio, requiriendo únicamente la obtención de determinada información presente en un archivo.

El proceso para utilizar este tipo de herramientas consta de cuatro fases (Figura E.6). La primera fase es la definición de la gramática. Esta vendrá habitualmente expresada en notación BNF o EBNF. La segunda fase es la definición de los posibles *tokens* léxicos. Las reglas de una gramática están formadas por símbolos no terminales y símbolos terminales. Los primeros invocan a reglas de la gramática; los segundos son los *tokens* que el analizador léxico extraerá, para lo cual hay que indicarle las reglas que gobiernan su construcción. Entre estos *tokens* se encontrarán las palabras claves de la notación, los identificadores, símbolos de puntuación, etc.

La tercera fase es la construcción de la gramática aumentada. Se denomina así a la gramática que contiene acciones a realizar durante el análisis de un archivo. De esta forma, es posible ir almacenando en estructuras de memoria la información de interés contenida en el archivo. La cuarta fase es la generación del analizador a partir de la gramática aumentada y las definiciones léxicas construidas anteriormente.

Esta vía tiene varias ventajas. La primera es que la herramienta de generación del analizador ya construye la estructura sintáctica que impone la gramática, y lo hace de forma transparente al programador. La herramienta puede ofrecer mecanismos para la gestión de errores (detección, recuperación, etc.) durante la fase de análisis. La segunda

[2] Aquellas gramáticas que sólo necesitan mirar un *token* hacia delante se denominan gramáticas LL(1).

SPICUM servicio de publicaciones

ventaja es que este enfoque puede servir como base para el futuro diseño de un analizador semántico o un compilador para la notación.

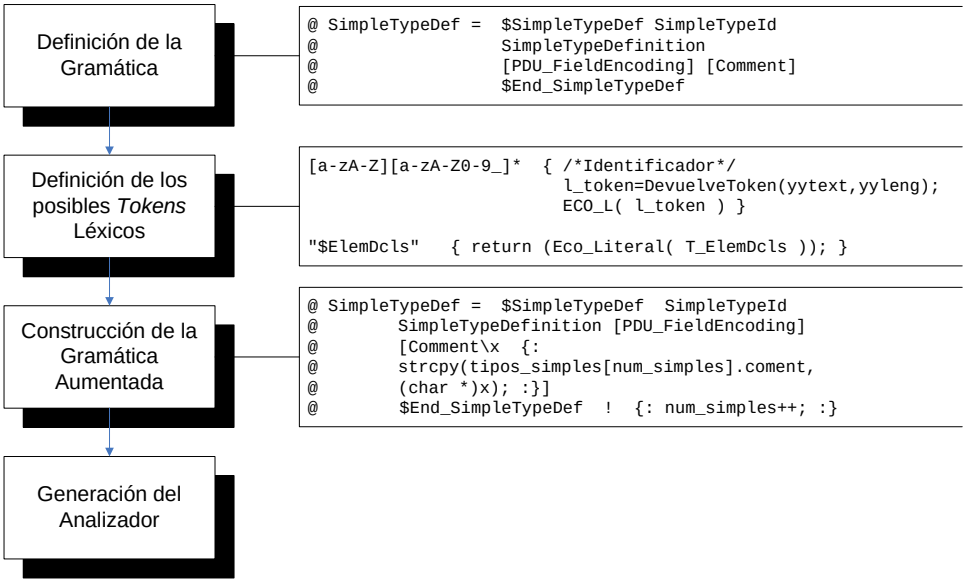


Figura E.6: Proceso de generación de un analizador sintáctico.

No obstante, también existen limitaciones. Inicialmente, el uso adecuado de este tipo de herramientas es más complejo y requiere un tiempo de aprendizaje. La construcción de la gramática aumentada puede requerir la implementación de estructuras de datos complejas y la funcionalidad necesaria para su gestión y utilización. Por ejemplo, tablas para almacenamiento de símbolos, árboles de orden 'n', pilas de símbolos para poder retroceder, etc. A pesar de ello, la potencia del resultado suele compensar este esfuerzo adicional inicial.

Aunque no se ha indicado al mencionar anteriormente las fases, puede ser necesario modificar algunas reglas de producción con objeto de que el generador del analizador trabaje adecuadamente. Por ejemplo, dependiendo de la herramienta usada, ante una alternativa hay que anteceder la de mayor o la de menor longitud, según cómo el generador recorra el árbol de símbolos de la gramática. Por lo general, esta modificación afectará a pocas reglas.

E.2.1 Tipos de Análisis

Como indica [GARS01], la forma más fácil de trabajar para un analizador sintáctico es mediante un análisis LL (*Left to right, Leftmost derivation*) (análisis *top-down*). Al leer un símbolo, se compara con las posibles reglas de producción y se decide cuál de ellas ha sido usada. Esto funciona si no hay dos posibles reglas de producción válidas. En este caso, es necesario mirar uno o más símbolos futuros hasta que sea posible decidirse por una u otra regla de producción. Se habla así de analizadores LL(0), LL(1), etc., según el número de símbolos que pueden considerar en avance.

La otra opción es utilizar un enfoque *bottom-up*. En este caso, el analizador va leyendo la entrada hasta que puede reducir los símbolos leídos en una sola regla. Existen diversas variantes, siendo las más potentes los algoritmos LALR (*Look Ahead Left to Right*) y LR(1) (*Left to right, Rightmost derivation*); el primero no es tan potente como

SPICUM servicio de publicaciones

el segundo, pero requiere mucho menos espacio de memoria. En [GARS01] se ofrece una comparativa entre los analizadores LL y LR (Tabla E.1); la figura muestra cuál de ellos ofrece mejores resultados.

Tabla E.1: Comparativa entre tipos de análisis.

	Análisis LL	**Análisis LR**
Simplicidad	✓	-
Generalidad	-	✓
Inclusión de acciones	✓	-
Memoria	✓	-

E.2.2 Herramienta *PRECCX*

Existen varias herramientas que permiten generar un analizador sintáctico a partir de las reglas de producción BNF. Se ha estudiado la viabilidad de utilizar *yacc* (*Yet Another Compiler Compiler*) ([LEVI92], [JOHN95], [BHAM98]), *ANTLR*[3] (*ANother Tool for Language Recognition*) ([ANTL95], [ANTL99]) y *PRECCX* (*PRE-C-Compiler eXtended*) ([BREU95], [BREU97]). Finalmente, se ha optado por utilizar la última de las tres ya que *yacc* no admite la notación EBNF, sino sólo reglas BNF; esto obliga a modificar las reglas incluidas en la especificación de TTCN [X.292] complicando sin necesidad el proceso de obtención de un conjunto de reglas de producción adecuado (Figura E.7). *ANTLR* se descartó porque el código resultante no era ANSI C (*American National Standards Institute* C), y se ha buscado la portabilidad de la herramienta final. Por estos motivos, la opción elegida ha sido *PRECCX*.

Regla EBNF

```
TS_ParDcls   ::=   $Begin_TS_ParDcls [TS_ParGroupRef] {[CollComment]
                   TS_ParDcl}+ [Comment] $End_TS_ParDcls
```

Reglas BNF equivalentes

```
TS_ParDcls   ::= $Begin_TS_ParDcls regla_aux_01 $End_TS_ParDcls

regla_aux_01 ::= TS_ParGroupRef regla_aux_02
                 | regla_aux_02 Comment
                 | TS_ParGroupRef regla_aux_02 Comment

regla_aux_02 ::= regla_aux_03
                 | regla_aux_03 regla_aux_02

regla_aux_03 ::= TS_ParDcl
                 | CollComment TS_ParDcl
```

Figura E.7: Ejemplo de conversión de una regla de producción EBNF en un conjunto de reglas BNF.

[3] Anteriormente denominado *PCCTS* (*Purdue Compiler Construction Tool Set*).

SPICUM servicio de publicaciones

PRECCX es un compilador de compiladores para gramáticas dependientes del contexto, que utiliza un análisis LL(∞). Al igual que *yacc*, *PRECCX* requiere un analizador léxico que obtenga los símbolos que forman la gramática; aunque incluye un analizador por defecto, permite que el usuario defina qué símbolos le son de interés. Si no se hace nada más, la salida de *PRECCX* será un código que admita como entrada un fichero e indique si está correctamente escrito o no.

SPICUM servicio de publicaciones

SPICUM
servicio de publicaciones

Apéndice

F

APÉNDICE F: DESCRIPCIÓN DEL SISTEMA DECT

DECT (*Digital Enhanced Cordless Telecommunications*) es un sistema digital microcelular de comunicación radio que ofrece un acceso inalámbrico de baja potencia entre los equipos móviles y las estaciones base (Figura F.1) con alcances de hasta varios cientos de metros [ETS 300 175-1]. La tecnología DECT está enfocada a la prestación de servicios inalámbricos en oficinas y entornos residenciales, permitiendo, desde sus inicios, tanto el envío de voz como de datos.

En este Apéndice se describe la Interfaz Común (CI – *Common Interface*) ([ETS 300 175], partes 1 a 8), que especifica la arquitectura y funcionalidad para que los servicios de voz y datos de equipos distintos puedan operar conjuntamente.

F.1 Características Técnicas

Un sistema DECT está compuesto por dos tipos de entidades[1]:

- Terminación Fija (FT – *Fixed Termination*): hace las funciones de estación base.
- Terminación Portátil (PT – *Portable Termination*): hace las funciones de equipo móvil.

El sistema DECT emplea, como técnicas de acceso, tanto la división en frecuencia como la división en tiempo. Opera en la banda de 1880 a 1900 MHz con 10 portadoras separadas 1,728 MHz entre sí[2]. La asignación de canales se realiza de forma dinámica, evitando el coste asociado a la planificación de frecuencias requerido en los sistemas celulares. El sistema también ofrece la realización de traspasos de las llamadas en curso tanto dentro de la propia célula como entre células diferentes.

[1] Las Especificaciones de Sistema distinguen entre Terminación (Fija o Portátil) y Parte (Fija o Portátil). Una Terminación incluye elementos definidos en [ETS 300 175], es decir, la funcionalidad del Nivel 1 y una selección de los Niveles 2 y 3. Una Parte contiene todos los elementos de la red DECT entre la red local y la interfaz aire DECT (Fija) o entre el usuario y la interfaz aire DECT (Portátil). Para evitar confusiones, y dado que el Sistema de Pruebas se refiere exclusivamente a funcionalidades incluidas en [ETS 300 175], se ha utilizado el nombre Terminación, Fija (FT) o Portátil (PT), en todo el documento.

[2] La frecuencia 0 es la más alta (1897,344 MHz) y la 9 es la más baja (1881,792 MHz).

SPICUM servicio de publicaciones

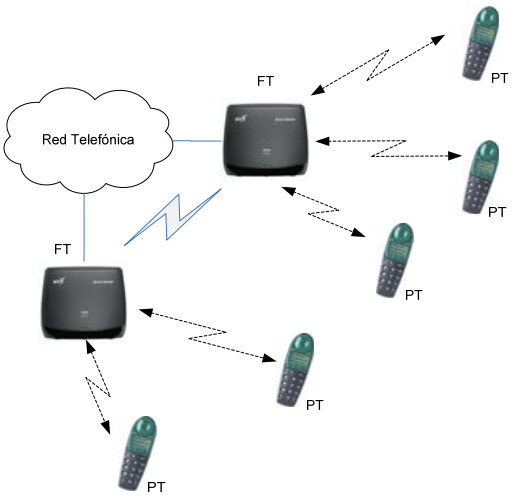


Figura F.1: Elementos del Sistema DECT.

Cada portadora ofrece una trama de duración 10 ms con 24 intervalos, empleándose las 12 (0-11) primeras en el sentido descendente FT→PT y las 12 (12-23) últimas en el sentido ascendente PT→FT. Con este esquema (Figura F.2), cada Terminación Fija dispone de 120 canales dúplex. Los dos intervalos que forman un canal dúplex se encuentran separados 5 ms o, equivalentemente, 12 intervalos. Se pueden emplear medios intervalos, intervalos completos e intervalos dobles. En un nivel superior, el estándar define lo que llama multitrama, una agrupación de 16 tramas básicas. La información de señalización y control presente en cada trama depende de su posición dentro de la multitrama

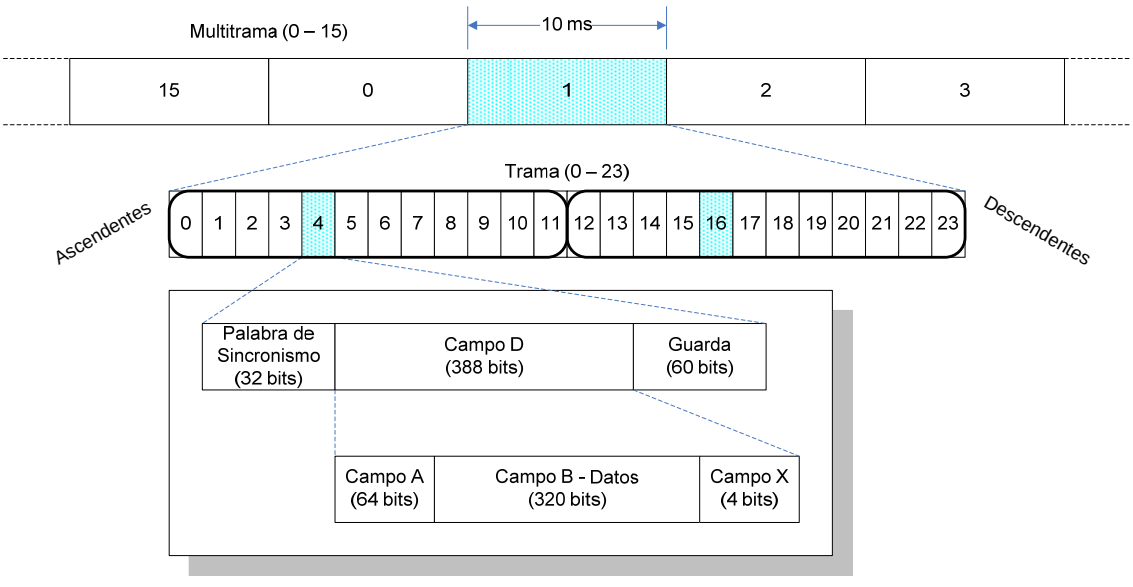


Figura F.2: Estructura de trama y multitrama.

Cada intervalo tiene una duración de 416,6 μs, ofreciendo la transmisión de un total de 480 bits: 420 bits (o 424 si se emplea el campo de detección de colisión, campo Z) de datos y un período de guarda de 60 bits. De los 420 bits de datos, realmente sólo 320 se encuentran disponibles para datos de usuario (campo B), empleándose el resto en una

SPICUM servicio de publicaciones

palabra de sincronismo (32 bits), información de control (campo A, 64 bits) y un código de redundancia cíclica (campo X, 4 bits). Con un intervalo completo, el usuario dispone de una velocidad de 32 kbps; juntando 2 intervalos se dispone del equivalente a un canal ISDN de 64 kbps[3].

F.2 Arquitectura de Protocolos

El sistema DECT se basa en la arquitectura plasmada en el modelo OSI [ISO 7498]. La torre de protocolos DECT consta de cuatro niveles (Figura F.3) que equivalen a los 3 primeros niveles OSI; además, existe un nivel de gestión que se comunica con los cuatro niveles anteriores. Por tanto, un equipo DECT está formado por los siguientes niveles:

- Nivel Físico (PHL – *PHysical Layer*).
- Nivel de Control del Acceso al Medio (MAC – *Medium Control Access*).
- Nivel de Control del Enlace (DLC – *Data Link Control*).
- Nivel de Red (NWK – *NetWorK*).
- Entidad de Gestión de los Niveles Inferiores (LLME – *Lower Layer Management Entity*).

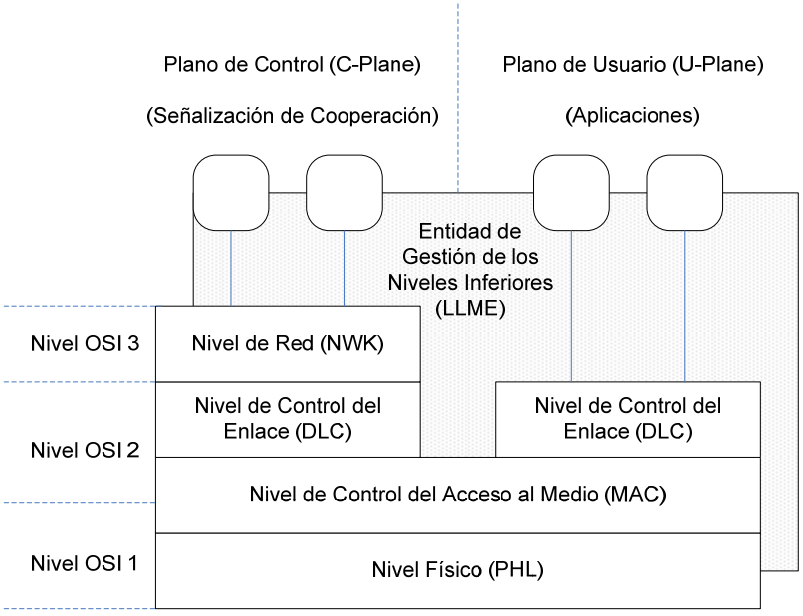


Figura F.3: Arquitectura de un equipo DECT [ETS 300 175-1].

La correspondencia entre modelos que aparece en la figura es meramente ilustrativa con el objeto de indicar la similitud entre uno y otro. Según las Especificaciones de Sistema, la diferente subdivisión en niveles es necesaria para poder tener en cuenta las incertidumbres generadas por la interfaz radio y el concepto de traspaso. El estándar

[3] Un canal DECT se puede emplear para trasmitir voz comprimida mediante ADPCM o para transmitir datos a una velocidad variable en función del número de ranuras asignadas a la comunicación.

SPICUM servicio de publicaciones

ofrece mecanismos en todos los niveles para permitir el uso de protocolos o características diferenciadores por parte de los fabricantes. Esto representa uno de los puntos clave para la evolución del sistema, como se ha demostrado en estos años [DEWE07].

Verticalmente, a partir del Nivel DLC se puede dividir la torre de protocolos en dos, que en el estándar se denominan planos:

- Plano de Control (*C-Plane*): contiene la información interna de control del protocolo. El Nivel de Red sólo existe en este plano.
- Plano de Usuario (*U-Plane*): contiene la mayoría de la información de usuario y de control extremo a extremo, sin incluir ninguna información interna de control del protocolo DECT.

A continuación se describen las funciones básicas de cada nivel presente en un equipo DECT.

F.2.1 Nivel Físico

El Nivel Físico (PHL – *PHysical Layer*) proporciona enlaces punto a punto entre dos terminales radio. Para ello, divide el espectro radio en las dimensiones de frecuencia y tiempo, empleando 10 portadoras con acceso TDMA (*Time Division Multiple Access*). Este nivel interactúa con el nivel MAC y la Entidad de Gestión. Sus funciones [ETS 300 175-2] se agrupan en:

- Modular y demodular las portadoras radio.
- Adquirir y mantener la sincronización de bit e intervalo.
- Transmitir y recibir una cantidad de bits definida en el instante y frecuencia indicados.
- Añadir y eliminar el campo de sincronización y el campo Z[4].
- Informar sobre la potencia de la señal recibida.

El nivel físico puede utilizar cuatro tipos de paquetes (Figura F.4):

a) Paquete Corto (P00, 96 bits): utilizado en la portadora piloto y en conexiones de datos.

b) Paquete Básico (P32, 420 bits): utilizado en la mayoría de los tipos de conexiones.

c) Paquete de Baja Capacidad (P08j, 180 bits): paquete que utiliza sólo la mitad de un intervalo.

d) Paquete de Alta Capacidad (P80, 900 bits): sólo se puede transmitir en intervalos pares.

El Paquete Corto (P00) debe ser soportado obligatoriamente; además, se debe soportar al menos uno de los otros tres tipos de paquetes. Opcionalmente, los paquetes P32, P08j y P80 pueden incluir el campo Z (4 bits adicionales).

[4] El campo Z (4 bits) se sitúa opcionalmente al final de cualquier paquete y permite detectar posibles colisiones.

SPICUM servicio de publicaciones

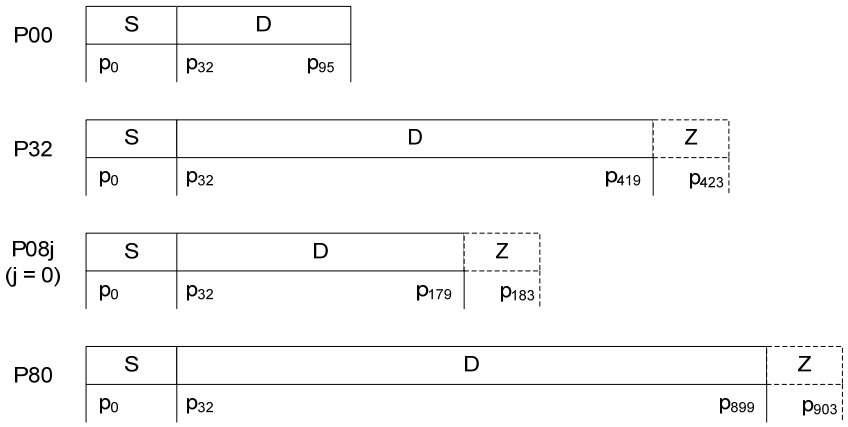


Figura F.4: Formato de los distintos tipos de paquetes del Nivel Físico.

F.2.2 Nivel de Control del Acceso al Medio

El Nivel de Control del Acceso al Medio (MAC – *Medium Access Control*) [ETS 300 175-3] permite seleccionar y establecer un canal físico, establecer y liberar conexiones en dichos canales y multiplexar la información de control con la información de los niveles superiores y la información de control de errores. El modelo de referencia de este nivel se muestra en la Figura F.5.

F.2.2.1 Arquitectura

Como se observa, la arquitectura de este nivel está dividida en dos bloques:

- Funciones de Control del Grupo (CCF – *Cluster Control Functions*): Contiene las funciones utilizadas para controlar más de una celda.
- Funciones de Celda del Emplazamiento (CSF – *Cell Site Functions*): Incluye todas las funciones relacionadas con una única celda.

En todo momento sólo hay una instancia de las funciones CCF, mientras que existirá una instancia de las funciones CSF por cada celda instalada. Esta posibilidad sólo existe en el caso de Terminaciones Fijas.

El Nivel MAC ofrece tres grupos de servicios, que se realizan mediante la funcionalidad contenida en el bloque CCF. Estos grupos de servicios, cada uno de los cuales se corresponde con un elemento funcional de control del bloque CCF, son los siguientes:

- Servicio de Difusión de Mensajes (BMC – *Broadcast Message Control*): Ofrece un servicio de comunicación no orientada a conexión punto a multipunto; se emplea fundamentalmente para difundir información sobre la Terminación Fija. Todas las portadoras transportan la misma información. Sólo hay una instancia del elemento funcional.
- Servicio de Mensajes No Orientados a Conexión (CMC – *Connectionless Message Control*): Proporciona servicios de comunicación no orientada a conexión punto a punto o punto a multipunto. Como mucho hay una instancia del elemento funcional.

SPICUM servicio de publicaciones

- Servicio de Multiportadora (MBC – *Multi-Bearer Control*): Ofrece servicios orientados a conexiones punto a punto, mediante una o más portadoras. Hay una instancia del elemento funcional para cada conexión.

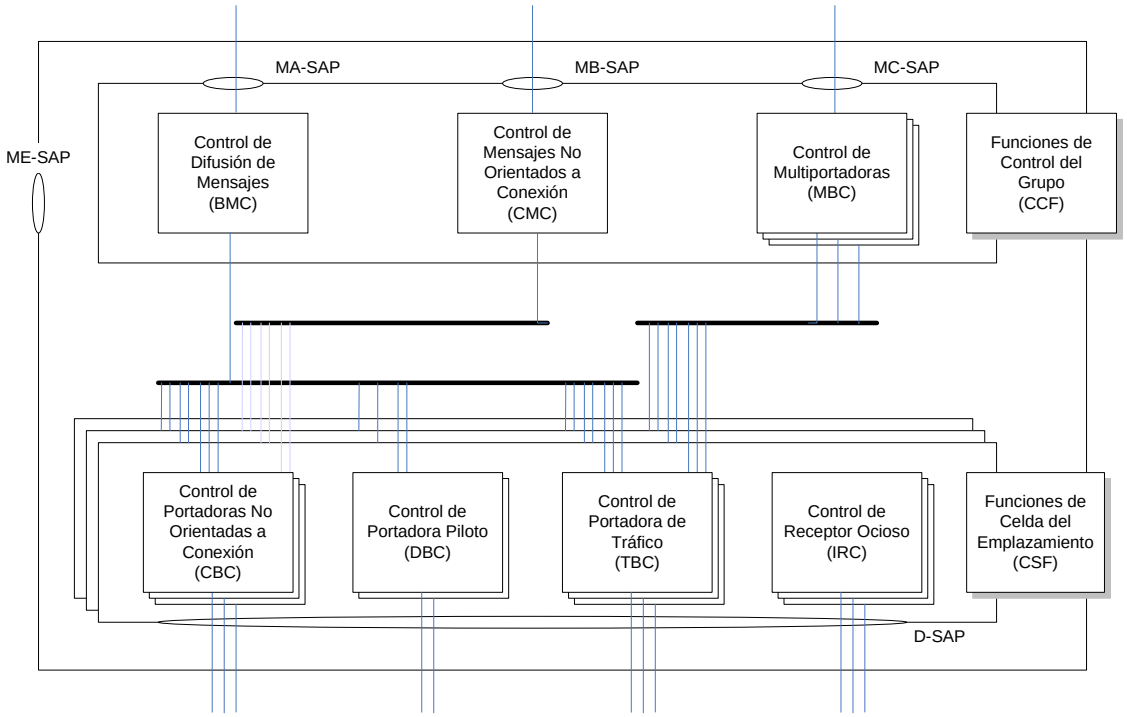


Figura F.5: Modelo de referencia del Nivel de Control del Acceso al Medio (MAC).

Por su parte, las funciones de Control de Celda del Emplazamiento (CSF) contienen los siguientes elementos funcionales de control:

- Control de Portadora No Orientada a Conexión (CBC – *Connectionless Bearer Control*).
- Control de Portadora Piloto (DBC – *Dummy Bearer Control*): un máximo de dos instancias en cada celda.
- Control de Portadora de Tráfico (TBC – *Traffic Bearer Control*): tantas instancias como portadoras de tráfico existan.
- Control de Receptor Ocioso (IRC – *Idle Receiver Control*): controla el receptor cuando no está ligado a una portadora, es decir, cuando no está siendo usado por una instancia de algún otro elemento funcional de control. Hay una instancia por cada transceptor.

F.2.2.2 Interfaz

La información de control y de usuario se organiza en canales lógicos (Tabla F.1) con dos propósitos: proporcionar servicios al nivel superior y transportar datos de control del propio nivel. Algunos de estos canales hacen uso de los Puntos de Acceso al Servicio del Nivel MAC, pero otro grupo, los canales lógicos internos, son propios del Nivel MAC y se utilizan para el envío de información importante para la comunicación entre diferentes entidades MAC.

SPICUM servicio de publicaciones

Los Puntos de Acceso al Servicio de que dispone el nivel MAC son los siguientes:

- Con el Nivel DLC: MA-SAP (Servicio de Difusión de Mensajes), MB-SAP (Servicio de Mensajes No Orientados a Conexión) y MC-SAP (Servicio de Multiportadora).
- Con el Nivel Físico: D-SAP.
- Con la Entidad de Gestión LLME: ME-SAP. La especificación sólo indica los servicios a prestar a través de este SAP, pero no define el conjunto de primitivas para ello.

Tabla F.1: Correspondencias entre función CCF, Punto de Acceso al Servicio y canales lógicos en el Nivel MAC.

Elemento Funcional	SAP	Plano	Canal	Descripción
BMC	MA-SAP	---	B_S	Canal simplex lento. Reduce la capacidad del canal lógico N.
CMC	MB-SAP	C	CL_S	Canal simplex lento. Reduce la capacidad del canal lógico N.
			CL_F	Canal simplex rápido. Mayor capacidad que el canal CL_S.
		U	SI_N	Adecuado para voz.
			SI_P	Datos protegidos con CRC de 16 bits.
MBC	MC-SAP	C	C_S	Canal dúplex lento. Reduce la capacidad del canal lógico N.
			C_F	Canal dúplex rápido. Reduce la capacidad de los canales I.
		U	G_F	Canal simplex rápido para controlar entidades del plano U.
			I_N	Incompatible con I_P. Adecuado para voz.
			I_P	Incompatible con I_N. Adecuado para datos.
Canales Lógicos Internos	---	---	M	Transmite información de control del Nivel MAC. Lleva los mensajes de establecimiento y liberación de portadoras.
			N	Canal de identidades. Transmite la identidad del sistema.
			P	Mensajes de aviso (*paging*). Puede incluir datos del canal B_S.
			Q	Canal de información del sistema. Canal simplex con información sobre la Terminación Fija de la celda.

F.2.2.3 Portadoras

El Nivel MAC proporciona tres tipos de portadoras:

- Simplex: asignación de un canal físico en un sentido. Puede ser una portadora simplex corta (sólo lleva el campo A) o larga (transmite mensajes con campo A y campo B).
- Dúplex: se forma como la unión de dos portadoras simplex, una en cada sentido. Estas portadoras deben estar equiespaciadas en la trama DECT.
- Simplex Doble: unión de dos portadoras simplex en el mismo sentido.

Cada portadora puede funcionar en uno de tres estados:

- Piloto: transmisión continua de información de difusión.
- Tráfico: transmisión continua punto a punto. No es válido para una portadora simplex.
- No Orientada a Conexión: para servicios no orientados a conexión y servicios de difusión.

SPICUM servicio de publicaciones

El orden de transmisión de la información, y los convenios para numerar los bits y octetos, aparecen en la Figura F.6.

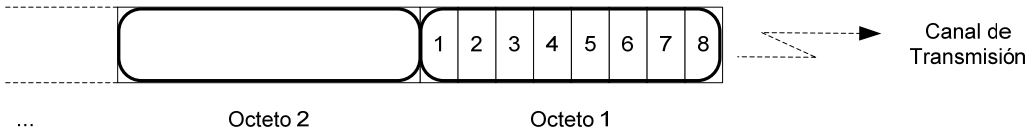


Figura F.6: Orden de transmisión de los mensajes MAC.

F.2.2.4 Identificación de Conexiones

El Nivel MAC ofrece dos tipos de conexiones: básicas y avanzadas. En las conexiones básicas, PT y FT no comparten un identificador común de la conexión; por este motivo, sólo puede haber una conexión básica entre un par FT-PT[5]. En las conexiones avanzadas, sin embargo, PT y FT comparten el ECN (*Exchanged Connection Number*), asignado por la Entidad de Gestión.

Localmente, cada conexión se identifica en el elemento funcional MBC mediante el MCEI (*MAC Connection Endpoint Identification*). En el caso de conexiones avanzadas, el identificador (ver Apartado F.3) es la concatenación del ARI (*Access Rights Identity*, identificador de la FT), el PMID (*Portable part MAC Identity*, identificador del PT) y el ECN.

F.2.2.5 Multiplexación

Con objeto de acomodar la información de todos los canales lógicos en la capacidad disponible, el Nivel MAC realiza dos tipos de multiplexación:

- Espacial: combina dos o más campos de datos en una sola trama. Son reglas para construir el campo D de un mensaje.
- Temporal: en el mensaje MAC se incluye información de un canal lógico u otro dependiendo de la posición de la trama en la estructura de multitrama.

Hay varios tipos de multiplexores temporales; de interés son el Multiplexor de Cola (T-MUX – *Tail MUltipleXer*) y el Multiplexor de campo B (E/U-MUX). El esquema de multiplexación se describe en la Sección 6 de [ETS 300 175-3]. El primer multiplexor maneja información de los canales lógicos P, Q, N, M y C (C_S, CL_S), que se mapea en el campo T (*tail*) del campo A de los mensajes. La prioridad de estos mensajes, así como su asignación temporal, se muestra en la Tabla F.2.

Por su parte, el multiplexor de campo B, E/U-MUX, permite decidir el contenido del campo B entre las opciones a) Tipo E: canales lógicos M, C_F, CL_F o G_F y b) Tipo U: canales lógicos I_N, I_P, SI_N.

[5] Esta condición se relaja en los procedimientos de traspaso, donde puede haber dos conexiones básicas activas simultáneamente.

SPICUM servicio de publicaciones

Tabla F.2: Prioridad de mensajes en el campo T del campo A (multiplexor T-MUX).

FT				PT			
Trama	**Prioridad[6]**	**Trama**	**Prioridad[6]**	**Trama**	**Prioridad[6]**	**Trama**	**Prioridad[6]**
0	P_T, N_T	1	M_T, C_T, N_T	0	M_T, C_T, N_T	1	N_T
2	P_T, N_T	3	M_T, C_T, N_T	2	M_T, C_T, N_T	3	N_T
4	P_T, N_T	5	M_T, C_T, N_T	4	M_T, C_T, N_T	5	N_T
6	P_T, N_T	7	M_T, C_T, N_T	6	M_T, C_T, N_T	7	N_T
8	Q_T	9	M_T, C_T, N_T	8	M_T, C_T, N_T	9	N_T
10	P_T, N_T	11	M_T, C_T, N_T	10	M_T, C_T, N_T	11	N_T
12	P_T, N_T	13	M_T, C_T, N_T	12	M_T, C_T, N_T	13	N_T
14	N_T	15	M_T, C_T, N_T	14	M_T, C_T, N_T	15	N_T

F.2.3 Nivel de Control del Enlace

El Nivel de Control del Enlace (DLC – *Data Link Control*) [ETS 300 175-4] es el encargado de proporcionar enlaces de datos fiables al Nivel de Red. Este Nivel se encuentra dividido en dos planos de operación:

- Plano C (*C-Plane*): es común a todas las aplicaciones, proporcionando enlaces punto a punto fiables para la transmisión de señalización de control interna, junto con una capacidad limitada para tráfico de usuario, y enlaces punto a multipunto para difusión de información.
- Plano U (*U-Plane*): ofrece un conjunto de servicios cuyas características están adaptadas a tipos específicos de servicios de transmisión de información de usuario.

F.2.3.1 Plano de Control

El Plano de Control (Plano C) proporciona dos servicios independientes: a) Servicio de Enlace de Datos y b) Servicio de Difusión. El modelo de referencia para las entidades que prestan estos servicios en el Plano de Control del Nivel DLC se muestra en la Figura F.7.

El Servicio de Enlace de Datos hace uso de las entidades LAPC y Lc. Con esta división en dos entidades, el diseño ha intentado separar las funciones de acceso típicas (LAPC) de funciones de control más particulares de este sistema (Lc). Existen una instancia LAPC y otra Lc por cada enlace de datos.

La entidad LAPC implementa un protocolo de enlace derivado del protocolo LAPD (*Link Access Protocol on D channel*)[7] de ISDN ([Q.920], [Q.921]); incluye mecanismos para detección y recuperación de errores y para el control de flujo. Una entidad LAPC se identifica por el Identificador del Enlace de Datos (DLI – *Data Link Identifier*).

La entidad Lc se encarga de encolar y fragmentar las tramas LAPC antes de cedérselas al Nivel MAC; en recepción recompone las tramas LAPC. Esta entidad se encarga,

[6] El orden implica prioridad en caso de haber información disponible en más de un canal lógico válido.

[7] Las diferencias entre el protocolo LAPC (*Link Access Procedure for Control plane*) empleado y el protocolo LAPD de ISDN consisten, básicamente, en incluir un campo de longitud en cada trama y discretizar las longitudes permitidas.

SPICUM servicio de publicaciones

además, de la generación y comprobación del *checksum*, de enrutar las tramas al canal lógico adecuado (C_S o C_F) y del traspaso de las conexiones.

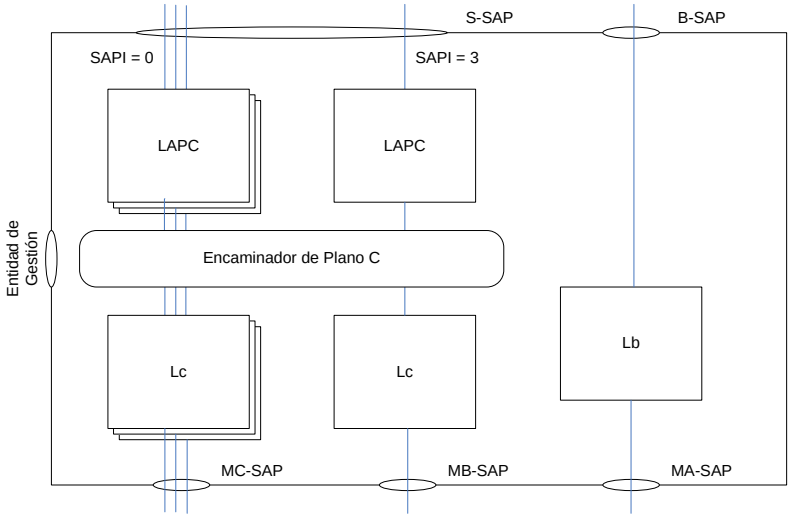


Figura F.7: Modelo de referencia del Plano de Control del Nivel de Control del Enlace (DLC).

Cada par LAPC – Lc puede proporcionar una de las tres siguientes clases de servicio:

- Servicio No Confirmado (Clase U)
- Servicio Confirmado
    - Una Trama (Clase A)
    - Múltiples Tramas (Clase B)

La comunicación con el Nivel de Red se realiza a través del Punto de Acceso al Servicio S-SAP. Este servicio se comunica con el Nivel MAC a través de los Puntos de Acceso al Servicio MB-SAP y MC-SAP. La información de Nivel de Red se encamina por uno u otro SAP del Nivel MAC según el parámetro SAPI (*Service Access Point Identifier*) de sus primitivas; para la señalización orientada a conexión se utiliza el valor `SAPI = 0`, y para la señalización no orienta a conexión se utiliza el valor `SAPI = 3`.

El Servicio de Difusión utiliza la entidad Lb. Esta entidad proporciona un servicio de difusión con tramas de longitud fija. Se comunica con el Nivel MAC a través del Punto de Acceso al Servicio MA-SAP y con el Nivel de Red a través del Punto de Acceso al Servicio B-SAP. Sólo hay una instancia de la entidad Lb.

El servicio de transferencia confirmada hace uso de las tramas I (*Information*), para enviar los datos, y RR (*Receive Ready*), para confirmar. La transferencia no confirmada emplea las tramas UI (*Unnumbered Information*).

F.2.3.2 Plano de Usuario

El Plano de Usuario (Plano U) proporciona distintos servicios de transferencia de información de usuario extremo a extremo, denominados en conjunto LUX[8]. El servicio básico es un servicio transparente no protegido adecuado para voz. Cada uno de estos

[8] El carácter 'X' se sustituye por un valor entre 1 y 16 según el servicio (ej: LU1, LU2, …, LU16).

SPICUM servicio de publicaciones

servicios es independiente y se accede a través de Puntos de Acceso al Servicio también independientes (LUX-SAPs). Cada servicio se desglosa en una entidad superior (LUx) y una entidad inferior (FBx), que incluyen, respectivamente, la funcionalidad dependiente del servicio y la funcionalidad de encolado, fragmentación y recomposición de tramas del Plano U. Los servicios del Plano U hacen uso de los canales lógicos I_N o I_P del Nivel MAC.

F.2.4 Nivel de Red

El Nivel de Red[9] (NWK – *NetWorK*) [ETS 300 175-5] proporciona la funcionalidad necesaria para el control de la llamada y del enlace, los servicios de mensajes orientados a conexión y no conexión, los servicios suplementarios y la gestión de la movilidad. A diferencia de los niveles inferiores, el Nivel de Red sólo existe en el Plano de Control[10]; el modelo de referencia correspondiente se muestra en la Figura F.8.

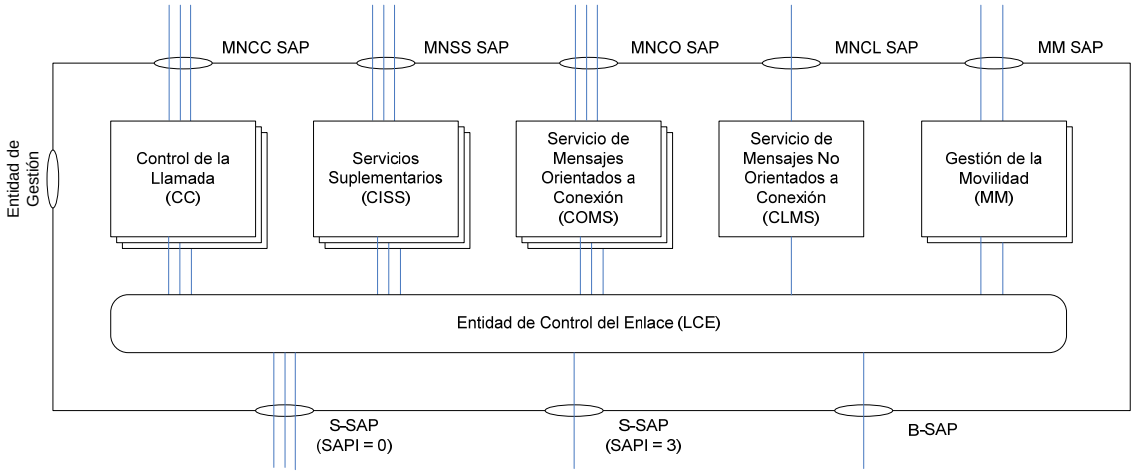


Figura F.8: Modelo de referencia del Plano de Control del Nivel de Red (NWK).

La funcionalidad del Nivel de Red está agrupada en las siguientes entidades:

- Control de la Llamada (CC – *Call Control*): Se encarga de establecer, mantener y liberar las llamadas en modo circuito conmutado y su señalización asociada. Proporciona el Punto de Acceso al Servicio MNCC.
- Servicios Suplementarios (CISS – *Call Independent Supplementary Services*): Funcionalidades adicionales no relacionadas con una instancia específica de Control de la Llamada. Proporciona el Punto de Acceso al Servicio MNSS.
- Servicio de Mensajes Orientados a Conexión (COMS – *Connection Oriented Message Service*): Ofrece un servicio punto a punto de paquetes orientado a conexión. Dispone de un establecimiento de llamada más eficiente que la entidad CC; así como de la posibilidad de suspender y reanudar la llamada. Proporciona el Punto de Acceso al Servicio MNCO.

[9] El Nivel de Red es el nivel principal de señalización.

[10] En el Plano de Usuario, para cada Punto de Acceso al Servicio LUX del Nivel DLC, existe un Punto de Acceso al Servicio NUX en el Nivel de Red.

SPICUM servicio de publicaciones

- Servicio de Mensajes No Orientados a Conexión (CLMS – *ConnectionLess Message Service*): Ofrece un servicio no orientado a conexión punto a punto o punto a multipunto con mensajes de longitud fija (FT→PT) o longitud variable (FT↔PT). Proporciona el Punto de Acceso al Servicio MNCL.
- Gestión de la Movilidad (MM – *Mobility Management*): Se encarga de la gestión de identidades y la autentificación, localización y registro en una celda; además, coordina las operaciones de Nivel de Red entre diferentes entidades y con los niveles inferiores. Proporciona el Punto de Acceso al Servicio MM. Una Terminación puede iniciar en cualquier momento un procedimiento de esta entidad. Para evitar posibles interbloqueos estos procedimientos están clasificados en tres niveles de prioridad[11].
- Entidad de Control del Enlace (LCE – *Link Control Entity*): Esta entidad se encarga de establecer, operar y liberar un enlace del plano C entre una FT y cada PT, ofreciendo un servicio de enrutamiento a las entidades anteriores[12]. Asigna los identificadores DLEI (*Data Link Endpoint Identifier*) a los puntos finales de la conexión.

Los mensajes que utiliza el Nivel de Red están clasificados en mensajes punto a punto, que emplean el formato S, y mensajes de difusión, que emplean el formato B. La información que transportan se agrupa en Elementos de Información (*Information Elements*); el orden de estos elementos en el mensaje está fijado por la especificación y su longitud puede ser fija o variable. Los mensajes con formato S siempre incluyen los siguientes Elementos de Información: a) el discriminador del protocolo, que identifica la entidad origen del mensaje; b) el identificador de transacción, que distingue las transacciones asociadas a una misma entidad y Terminación Portátil; y c) el tipo de mensaje.

F.2.5 Entidad de Gestión

La Entidad de Gestión de los Niveles Inferiores (LLME – *Lower Layer Management Entity*) contiene aquellas funciones que involucran a más de un nivel. El LLME da servicio a todos los niveles; así, en la Figura F.3, aparece como un plano que da soporte a todos y cada uno de los niveles. Es una forma de acceder a información que se encuentra distribuida por toda la arquitectura, soslayando las rigideces que impone el modelo OSI. Sus funciones en cada nivel son las siguientes (Figura F.9):

- PHL: Se encarga de fijar la sincronización de intervalo y de trama, recabar la lista de los canales con menos interferencias, e identificar los canales físicos con mayor potencia de señal.
- MAC: En el Nivel MAC, la Entidad de Gestión se encarga de la gestión de los canales físicos, y de la creación, mantenimiento y liberación de portadoras. Además, en difusión comunica a la entidad BMC la información necesaria

[11] Nivel de prioridad 1 (Autentificación de FT). Nivel de prioridad 2 (Terminación de los derechos de acceso, iniciada por FT; Autentificación de PT; Autentificación del usuario; Cifrado, iniciado por FT; Identificación de PT; Asignación de clave; Actualización de la localización; Asignación temporal de identidad). Nivel de prioridad 3: (Obtención de los derechos de acceso; Inscripción de localización).

[12] Este enrutamiento se realiza mediante el identificador DLEI (*Data Link Endpoint Identifier*) y el discriminador de protocolo de la entidad superior.

SPICUM servicio de publicaciones

(identidades PARI o SARI) para los canales N y Q en la FT (transmisión) y en sentido contrario en las PT (recepción).

- DLC: Lleva a cabo la gestión (establecimiento, liberación y traspaso) de las conexiones y del enrutamiento de los datos de los Planos C y U. Además, proporciona coordinación y control entre ambos planos, como en el caso de traspaso de conexiones.
- NWK: En el Nivel de Red gestiona la negociación y modificación de servicios de acuerdo con las peticiones del usuario, el uso de los recursos, la coordinación de los procedimientos de movilidad y el cifrado de las llamadas.

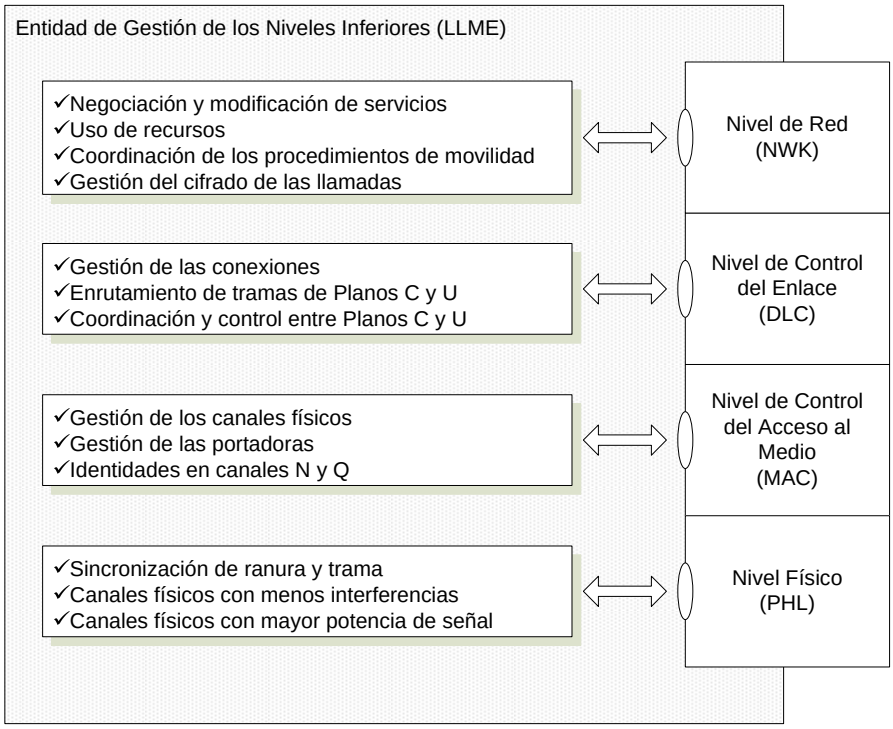


Figura F.9: Actividades de la Entidad de Gestión para cada nivel de la arquitectura DECT.

F.3 Identidades

El estándar DECT define una serie de identidades asociadas a las Terminaciones Fija (FT) y Portátil (PT) [ETS 300 175-6]. El objetivo es la protección del sistema frente a usos fraudulentos (no autorizados) y la prestación segura del servicio. Básicamente, una PT sólo puede acceder a una FT si alguna de las Identidades de Derechos de Acceso (ARIs – *Access Rights Identity*) de la Terminación Fija es válida para la identidad (PARK – *Portable Access Right Key*) de la Terminación Portátil.

La Tabla F.3 muestra todas las entidades correspondientes a un equipo DECT. Las identidades relacionadas con la conexión se utilizan para identificar las distintas instancias asociadas con una llamada. Las identidades relacionadas con el equipo son asignadas por el fabricante y lo identifican unívocamente; esta identidad se puede utilizar como autentificación en las llamadas de emergencia. En los apartados siguientes se explican las distintas variantes de identidades del Nivel de Red y su utilización.

SPICUM servicio de publicaciones

Tabla F.3: Lista de identidades utilizadas por equipos DECT.

	Nivel	Identidad	Nombre	Descripción
Identidades del Equipo	---	IPEI	*International Portable Equipment Identity*	Identificación unívoca globalmente; asignada por el fabricante. Está formada por otras dos identidades: EMC y PSN.
		EMC	*Equipment Manufacturer Code*	Identificación del fabricante. Son 16 bits.
		PSN	*Portable equipment Serial Number*	Número de serie de la Terminación Portátil. Son 20 bits.
Identidades Relacionadas con la Conexión	**MAC**	FMID	*Fixed part MAC Identity*	Los 12 bits menos significativos de la RFPI. Se usa para evitar interferencia co-canal.
		PMID	*Portable part MAC IDentity*	Son 20 bits, con un valor arbitrario o un TPUI asignado individualmente.[13]
	DLC	LSIG	*Link SIGnature*	Firma del enlace, generada a partir del PMID.
	NWK	ARI	*Access Rights Identity*	Utilizado en el establecimiento de llamada y para autentificación.
		IPUI	*International Portable User Identity*	Utilizado para *paging*, establecimiento de llamada y autentificación.
		TPUI	*Temporary Portable User Identity*	Más corto que el IPUI e iguales funciones; asignado en un dominio, pero sólo temporalmente.
Otras	---	RFPI	Radio Fixed Part Identity	Identifica RFPs (Radio Fixed Part) globalmente, transporta el PARI y marca los dominios.
		PARK	*Portable Access Rights Key*	Es el nombre del ARI de la Terminación Fija en la Terminación Portátil.

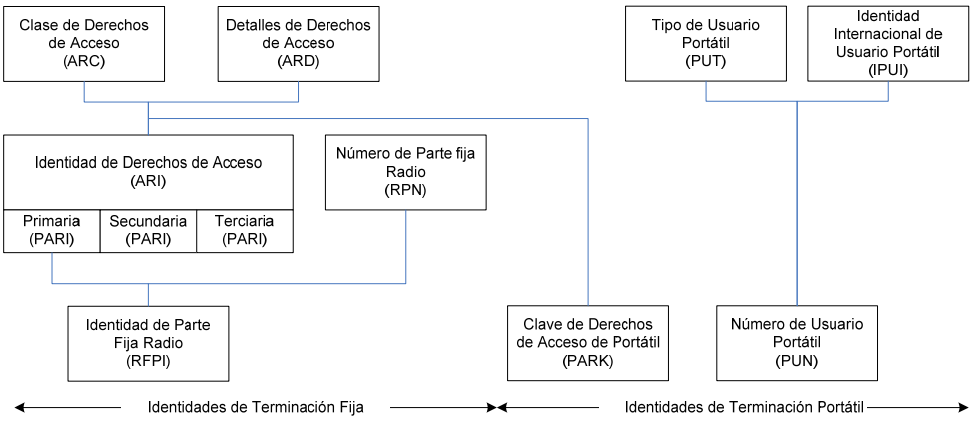


Figura F.10: Estructura general de las identidades DECT.

F.3.1 Identidades de Terminación Fija

La base de las identidades (Figura F.10) de una Terminación Fija son las Identidades de Derechos de Acceso (ARIs – *Access Rights Identity*). Una ARI se representa en 31 ó 36 bits. Existen tres tipos de ARIs:

[13] Los cuatro bits más significativos indican la opción elegida.

SPICUM servicio de publicaciones

- ARI Primaria (PARI – *Primary Access Rights Identity*): Se difunde periódicamente en el canal N_T, al menos una vez por multitrama.
- ARI Secundaria (SARI – *Secondary Access Rights Identity*): También se difunde, aunque menos frecuentemente que la Primaria. Se transmite en el canal Q_T. Sólo pueden ser ARIs de 31 bits.
- ARI Terciaria (TARI – *Tertiary Access Rights Identity*): Sólo está disponible bajo petición de la Terminación Portátil. Sólo pueden ser ARIs de 31 bits.

Cada ARI está formada por la concatenación de (Figura F.11):

- Clase de Derechos de Acceso (ARC – *Access Rights Class*): indica el tipo de acceso a la red DECT. Ocupa los 3 bits más significativos del ARI. Hay 4 clases para las Terminaciones Fijas (Tabla F.4).
- Detalles de Derechos de Acceso (ARD – *Access Rights Details*): número único dentro de un proveedor de servicio. Ocupa los 28 ó 33 bits menos significativos del ARI.

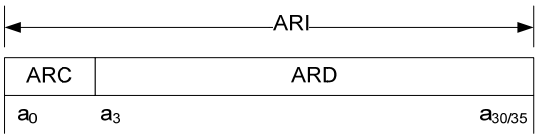


Figura F.11: Estructura de una Identidad de Derechos de Acceso (ARI).

La identidad de una Terminación Fija es la RFPI (*Radio Fixed Part Identity*). Esta identidad se difunde en el canal N_T, al menos una vez por multitrama; está formada por 40 bits (Figura F.12), que incluyen los siguientes campos:

- Bit E: Indica si la Terminación Fija dispone de identidades secundarias (SARIs)[14].
- Identidad Primaria (PARI): Identidad primaria de la Terminación Fija. Es globalmente única dentro de un operador.
- Número de la Terminación Fija Radio (RPN – *Radio fixed Part Number*): Valor utilizado para poder ofrecer varias celdas con la misma identidad primaria. Ocupa 8 ó 3 bits, según el tamaño del PARI.

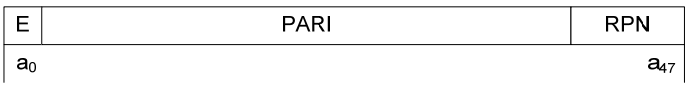


Figura F.12: Estructura de la Identidad de Terminación Fija RFPI.

[14] De la misma forma, el mensaje que transporta las identidades secundarias incluye también una indicación de si la Terminación Fija dispone de identidades terciarias (TARIs) o no.

SPICUM servicio de publicaciones

F.3.2 Identidades de Terminación Portátil

Cada Terminación Portátil se identifica (Figura F.10) por una Clave de Derechos de Acceso de Portátil (PARK – *Portable Access Rights Key*) y una Identidad Internacional de Usuario Portátil (IPUI – *International Portable User Identity*):

Tabla F.4: Clases de identidades para Terminaciones Fijas y Terminaciones Portátiles.

ARI		Entorno de Utilización	Uso de SARI/TARI	PARK		IPUI	
Clase	Bits			Clase	Bits	Tipo	Bits
A	36	Residencial y PABX privadas	No	A	36	N	40
						S	64
B	31	PABXs privadas	Sí	B	31	O	64
						S	64
						T	64
C	31	Servicio público	Sí	C	31	P	100
						Q	84
						S	64
						U	84
D	31	Acceso DECT a redes GSM	Sí	D	31	R	64

- PARK (*Portable Access Rights Key*): Esta clave es equivalente a la identidad ARI de la Terminación Fija, y define los derechos de acceso de la Terminación Portátil. Su estructura es como la de una ARI (Figura F.11), estando formado por la concatenación de un campo ARC (3 bits) y un campo ARD (28 ó 33 bits)[15]. Hay cuatro clases de PARK (campo ARC).
- IPUI (*International Portable User Identity*): Identifica unívocamente a la Terminación Portátil en el dominio definido por la ARI seleccionada. Un IPUI está formado por dos campos:
    - Tipo de Usuario Portátil (PUT – *Portable User Type*): define el plan de numeración utilizado.
    - Número de Usuario Portátil (PUN – *Portable User Number*): número unívoco dentro de un plan de numeración (PUT).

  DECT define siete tipos de IPUI para diferentes usos (Tabla F.4). Toda Terminación Portátil DECT dispone, al menos, de un IPUI de tipo N, asignado por el fabricante; este tipo de IPUI se utiliza para entornos residenciales, pero es también válido en cualquier entorno para llamadas de emergencias.

  Cuando una Terminación Portátil se registra en una Terminación Fija, ésta le puede asignar un identificador temporal, más corto que el IPUI, denominado Identidad Temporal de Usuario Portátil (TPUI – *Temporary Portable User Identity*); esta identidad temporal sólo es válida en dicha área de registro. La TPUI ocupa 20 bits[16].

[15] El parámetro Indicador de Longitud de *PARK* (PLI – *Park Length Indicator*) indica cuántos bits del *PARK* son significativos a la hora de compararlo con la identidad ARI de una Terminación Fija.

[16] Las identidades TPUI pueden ser individuales o de grupo; el tipo se indica en los 4 u 8 bits más significativos de la TPUI.

SPICUM servicio de publicaciones

Una Terminación Portátil puede acceder a cualquier Terminación Fija que posea un ARI (difunde PARI y SARIs, almacena TARIs) incluido en alguno de los PARKs de la Terminación Portátil; posteriormente, la Terminación Portátil debe identificarse válidamente en dicha Terminación Fija con su IPUI.

F.3.3 Ejemplo de Uso de las Identidades

Un posible escenario del uso de la estructura de identidades ARI de una Terminación Fija y las claves PARK de una Terminación Portátil es una red de servicio público (Figura F.13).

Las Terminaciones Fijas difunden su Identidad Primaria (PARI) en el canal N_T. Si el operador del servicio tiene acuerdos con otros operadores, difundirá los ARIs de dichos operadores como Identidades Secundarias (SARIs) en el canal Q_T. Una Terminación Portátil registrada en alguno de estos operadores identificará alguna de estas Identidades Secundarias como igual a su clave PARK, estando habilitada, por tanto, para acceder al servicio.

Si el número de Identidades Secundarias es excesivo para la capacidad del canal Q_T, algunas de estas identidades, las menos usadas, se pueden almacenar en la lista de Identidades Terciarias (TARIs). Una Terminación Portátil que visite este dominio podrá solicitar la comparación de sus PARKs con esta lista de Identidades Terciarias si no encuentra adecuada ni la Identidad Primaria ni ninguna de las Identidades Secundarias.

En el ejemplo mostrado en la Figura F.13, la Terminación Portátil no encuentra ninguna identidad válida en las Identidades Primarias y Secundarias difundidas por la Terminación Fija, por lo que solicita derechos de acceso enviando una de sus claves PARK y su IPUI asociado.

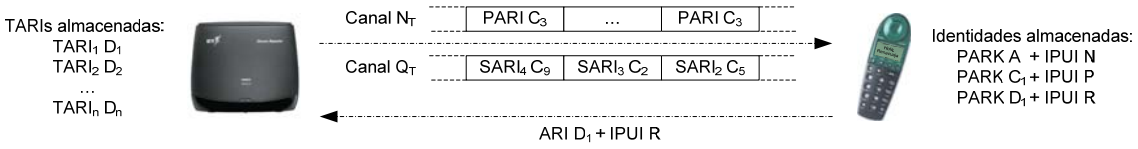


Figura F.13: Ejemplo de uso de Identidades de Derechos de Acceso en un escenario público.

F.4 Perfiles

DECT ha definido un conjunto de perfiles para aplicaciones específicas, incluyendo, desde su concepción, el acceso a otras redes públicas como GSM o ISDN. Las principales diferencias entre los distintos perfiles se encuentran en el ámbito de los protocolos. A todos ellos les son aplicables los requisitos radio definidos en [ETS 300 175-2]. La Tabla F.5 lista los perfiles que se encuentran definidos, junto con la norma correspondiente, una breve descripción de la funcionalidad que ofrecen y ejemplos de equipos comerciales.

El Perfil de Acceso Genérico (GAP – *Generic Access Profile*) [ETS 300 444] incluye la funcionalidad básica para aplicaciones de telefonía de voz con un ancho de banda de 3,1 kHz, estando enfocado hacia áreas como la telefonía doméstica o las centralitas de empresa. Este perfil es, a su vez, la base de otros perfiles que hacen uso de servicios de voz.

SPICUM servicio de publicaciones

El Perfil de Acceso Público (PAP – *Public Access Profile*) [ETS 300 175-9] especifica los requisitos mínimos para un equipo DECT que ofrezca o utilice servicios de acceso público. Este perfil ha sido retirado.

Tabla F.5: Perfiles DECT.

	Perfil	Norma	Servicio	Equipos Comerciales
---	Generic Access Profile (GAP)	[ETS 300 444]	Telefonía a 3,1 kHz.	DECT-8000 (Samsung), D500 ATEX (Philips), BS340 (Ericsson), One Touch First (ATLINKS)
	Cordless Terminal Mobility (CTM)	[EN 300 824]	Basado en GAP con traspaso entre redes públicas y privadas.	C4050 (Meridian)
	Integrated Services Digital Network (ISDN)	[EN 300 434] [EN 300 822]	Conectividad a redes ISDN.	BeeTel 455i (DeTeWe)
	GSM	[EN 300 370]	Conectividad a redes GSM.	DT570 (Ericsson)
	Terminales duales	[EN 301 242]	Integración de terminales duales DECT/GSM.	CeLInd (Unicom Pty)
	Radio in the Local Loop (RLL)	[EN 300 765]	Telefonía vía radio en el bucle local.	DRA 1900 (Ericsson)
	UMTS	[TS 101 863]	Conectividad a redes UMTS.	-- No encontrados --
	Acceso a redes IP	[TS 102 265]	Conectividad a redes IP.	DataGate HW 8660 (Höft & Wessel)
Data Service Profiles (DSP) [ETR 178]	DECT Packet Radio Service (DPRS)	[EN 301 649]	Paquetes de datos.	Basisstation (Yakumo)
	DECT Multimedia Access Profile (DMAP)	[EN 301 650]	Añade a GAP capacidades básicas de datos.	Gigaset S645 (Siemens)
	Ethernet	[TS 102 014]	Conectividad a redes LAN.	DSL600EVW (Aztech)
	V.24	[TS 102 012]	Conectividad vía modem V.24.	DECT 300 (Elmeg)
	D.2	[EN 301 238]	Servicio de velocidad adaptable para interconexión a interfaces V.x.	BP128i (Ericsson)
	Open Data Access Profile (ODAP)	[TS 102 342]	Aplicaciones industriales y residenciales a baja velocidad y con bajo consumo.	KIRK 5040 (Polycom)
	Low Rate Message Service (LRMS)	[EN 300 757]	Servicios de mensajes en redes públicas y privadas.	BT AirTalk (British Telecomm)
	Fixed line Multimedia Messaging Service (F-MMS)	[TS 102 379]	Mensajería multimedia para redes fijas.	Gigaset SL740 (Siemens)

DECT también ofrece una variada familia (Figura F.14) de Perfiles de Servicios de Datos (DSP – *Data Service Profiles*) ([ETR 178], [TR 102 185]), con objeto de proporcionar la funcionalidad mínima requerida por distintos usuarios para aplicaciones no de voz (Tabla F.5). En combinación con el GAP, este perfil admite el soporte de comunicaciones multimedia, de voz y de datos en la misma infraestructura. El perfil establece una división según el soporte de movilidad ofrecido: los servicios clase 1 son aplicaciones de área local; los servicios clase 2 son aplicaciones tanto públicas como privadas con posibilidad de cambio de celda sin requerir un pre-registro previo en la misma. Este conjunto de perfiles permite obtener tasas de transferencia de hasta 5 Mbit/s con tasas de error menores de 10^{-9}.

SPICUM servicio de publicaciones

Además, desde su concepción DECT ha contemplado el acceso a redes públicas de voz y datos. El perfil para acceso a redes públicas GSM está descrito en [EN 300 370]. Se basa en los servicios ofrecidos por el perfil GAP, a los que se añaden algunos otros requeridos por GSM, como por ejemplo, algoritmos adicionales de autentificación y cifrados. Para la realizar la interconexión, es necesario establecer una conexión entre una Terminación Fija y un centro de conmutación móvil (MSC: *Mobile Switching Centre*) GSM. Para ISDN (IAP – *ISDN Access Profile*) se han definido dos perfiles, correspondientes a un sistema final [EN 300 434] y un sistema intermedio [EN 300 822]. El primero está basado en GAP, suplementándolo con un servicio de datos a 64 kbit/s. Como sistema intermedio, el equipo DECT proporciona un enlace inalámbrico entre una red ISDN y uno o más terminales ISDN.

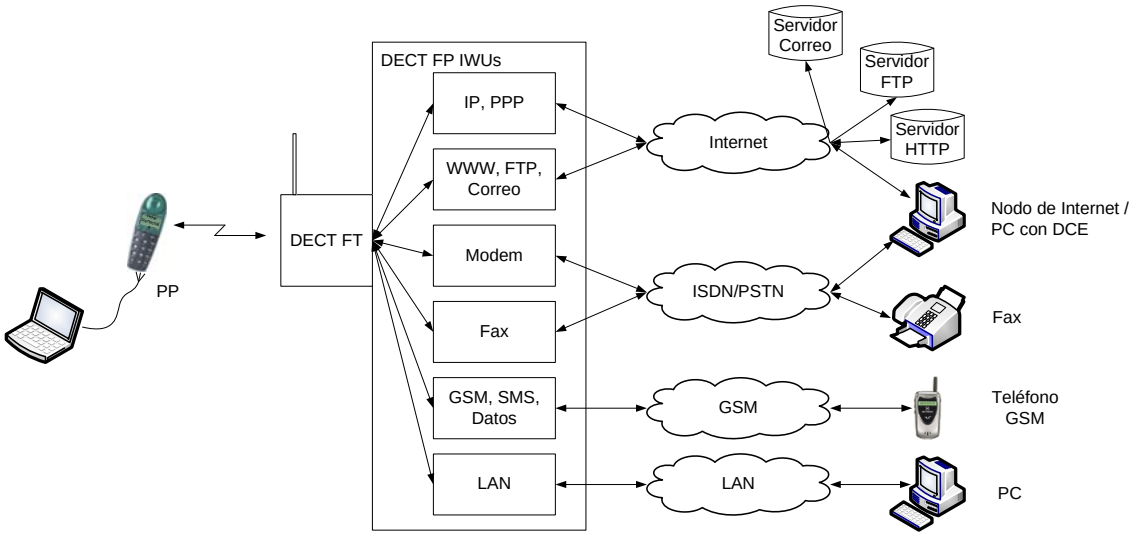


Figura F.14: Esquema resumen de los Perfiles de Servicios de Datos de DECT.

En 2007, ETSI ha empezado a publicar las especificaciones del DECT de Nueva Generación (*New Generation DECT*) ([TR 102 570], [TS 102 527]), que mejora la calidad de la comunicación y especifica el acceso a servicios Internet, por ejemplo, VoIP y *streaming* de video. Esta evolución está siendo comercializada bajo la denominación CAT-iq (*Cordless Advanced Technology—internet and quality*)[17]. Hay equipos ya disponibles como, por ejemplo, el modelo Icarus 1500 de Binatone.

F.4.1 Funcionalidad del Perfil GAP

El Perfil GAP incluye la funcionalidad que se indica en la Tabla F.6, la Tabla F.7 y la Asimismo, la Especificación del Perfil GAP establece limitaciones a los procedimientos de gestión. La Terminación Portátil iniciará el procedimiento de registro de localización, si el bit a38 difundido es '1', siempre que se produzca un cambio de Área de Localización y al encender la Terminación Portátil; si el bit es '0', sólo se inicia dicho procedimiento cuando es solicitado por la Terminación Fija. Además, sólo se asigna una identidad TPUI por suscripción, la cual se deriva del IPUI, si ha sido

[17] CAT-iq es el programa de certificación del DECT Forum para esta evolución del sistema.

SPICUM servicio de publicaciones

asignado, o del IPEI, en caso contrario. La identidad TPUI se borra al cambiar de Área de Localización. Otras limitaciones se pueden consultar en [ETS 300 444], Sección 13.

Tabla F.8 [ETS 300 444]. En esta tabla se indica si una característica o servicio es obligatorio (M), opcional (O), no se considera (I) o condicional (C); para los condicionales se indica la condición al final de la tabla correspondiente.

Tabla F.6: Características del Nivel NWK incluidas en el perfil GAP.

Item	Característica	PT	FT[18]	
			R/B	P
N.1	Llamada saliente.	M	M	M
N.2	Descolgar.	M	M	M
N.3	Colgar.	M	M	M
N.4	Marcación de dígitos (básica).	M	M	M
N.5	Registro de rellamada.	M	O	O
N.6	Cambio a señalización DTMF.	M	O	M
N.7	Pausa de marcación.	M	O	O
N.8	Llamada entrante.	M	M	M
N.9	Autentificación de PT.	M	O	M
N.10	Autentificación de usuario.	M	O	O
N.11	Registro de localización.	M	O	M
N.12	Asignación de clave en el aire.	M	O	O
N.13	Identificación de PT.	M	O	O
N.14	Indicación/Asignación de clase de servicio.	M	O	M
N.15	Alertas.	M	M	M
N.16	ZAP.	M	O	O
N.17	Activación de cifrado iniciada por FT.	M	O	M
N.18	Procedimiento de registro de suscripción en el aire.	M	M	M
N.19	Control del enlace.	M	M	M
N.20	Rescisión de derechos de acceso iniciada por FT.	M	O	O
N.21	Liberación parcial.	O	O	O
N.22	Cambio a DTMF (tono de longitud infinita).	O	O	O
N.23	Cambio a pulsos.	O	O	O
N.24	Señalización de caracteres visualizados.	O	O	O
N.25	Visualización de caracteres de control.	O	O	O
N.26	Autentificación de FT.	O	O	O
N.27	Activación de cifrado iniciada por PT.	O	O	O
N.28	Desactivación de cifrado iniciada por FT.	O	O	O
N.29	Desactivación de cifrado iniciada por PT.	O	O	O
N.30	Presentación de identificación de línea llamante (CLIP[19]) .	O	O	O
N.31	Llamada interna.	O	O	O
N.32	Llamada de servicio.	O	O	O
N.33	Conexión mejorada de Plano U.	O	O	O

Básicamente, en el Nivel de Red es capaz de realizar llamadas salientes, realizar llamadas entrantes, registrar su localización, identificarse ante la red, cifrar las comunicaciones y autentificar al usuario. La Especificación del Perfil especifica que:

- Todos los bits reservados en los mensajes del Nivel de Red se pondrán a '0'.

[18] Se distinguen los escenarios R/B ('*Residential/Business*') y P ('*Public*').

[19] *Calling Line Identification Presentation*.

SPICUM servicio de publicaciones

- Toda entidad (PT o FT) debe ser capaz de recibir mensajes de al menos 63 octetos (sólo requiere el uso de un segmento DLC).
- El identificador de transacción será el número más bajo de los disponibles.
- Simplifica la gestión de mensajes erróneos e inesperados[20].
- Los elementos de información `<<IWU-ATTRIBUTES>>` y `<<CALL-ATTRIBUTES>>` no tienen que ser procesados.

En el Nivel de Enlace, una entidad GAP puede utilizar, en el Plano de Control, los servicios Clase A (servicio confirmado) y Lb (servicio de difusión), y en el Plano de Usuario, procedimientos de Clase 0 con retardo mínimo, en concreto el servicio LU1 (servicio transparente no protegido). Solamente se manejan los canales lógicos de tipo C_S.

Tabla F.7: Servicios DLC para el perfil GAP.

Item	Servicio	PT	FT[18]	
			R/B	P
D.1	Servicio LAPC de Clase A y Lc.	M	M	M
D.2	Fragmentación y recombinación en canal Cs.	M	M	M
D.3	Servicio de difusión Lb.	M	M	M
D.4	Traspaso voluntario de conexión intra-celda.	M	C201	C201
D.5	Traspaso voluntario de conexión inter-celda.	M	O	O
D.6	Activación de cifrado.	M	C202	M
D.7	LU1 TRUP (*TRansparent UnProtected service*) Clase 0/retardo mínimo.	M	M	M
D.8	FU1.	M	M	M
D.9	Desactivación de cifrado.	C203	C203	C203
C201: Si servicio M.9, entonces O, en otro caso M. C202: Si característica N.17 o N.27, entonces M, en otro caso I. C203: Si característica N.29 o N.28, entonces M, en otro caso I.				

El Nivel de Acceso al Medio incluye la funcionalidad necesaria para realizar conexiones básicas, controlar la calidad del enlace, transmitir y recibir información de difusión, y, en el caso de la Terminación Portátil, realizar traspasos intra-celda e inter-celda, soporte de identidades SARI y activación del cifrado.

El Nivel Físico estipula que se deben utilizar intervalos completos, y que debe usarse el campo Z, de detección de colisión, en todos los paquetes. Una Terminación Fija debe poder escuchar y transmitir en las diez frecuencias y al menos en la mitad de los pares de intervalos 0 a 11; por su parte, una Terminación Portátil debe poder escuchar y transmitir en cualquier par de intervalos del 0 al 11.

Asimismo, la Especificación del Perfil GAP establece limitaciones a los procedimientos de gestión. La Terminación Portátil iniciará el procedimiento de registro de localización, si el bit a38 difundido es '1', siempre que se produzca un cambio de Área de Localización y al encender la Terminación Portátil; si el bit es '0', sólo se inicia dicho procedimiento cuando es solicitado por la Terminación Fija. Además, sólo se asigna una identidad TPUI por suscripción, la cual se deriva del IPUI, si ha sido

[20] La Sección 6.9.4 de [ETS 300 444] explica las acciones a tomar en cada caso. Por ejemplo, en caso de recepción de mensajes inesperados, estos serán ignorados salvo los mensajes {CC-RELEASE} y {CC-RELEASE-COM}.

SPICUM servicio de publicaciones

asignado, o del IPEI, en caso contrario. La identidad TPUI se borra al cambiar de Área de Localización. Otras limitaciones se pueden consultar en [ETS 300 444], Sección 13.

Tabla F.8: Servicios MAC para el perfil GAP.

Item	Servicio	PT	FT[18]	
			R/B	P
M.1	General.	M	M	M
M.2	Difusión continua.	M	M	M
M.3	Difusión de avisos.	M	M	M
M.4	Conexiones básicas.	M	M	M
M.5	Señalización de niveles altos por Cs.	M	M	M
M.6	Control de calidad.	M	M	M
M.7	Activación de cifrado.	M	C301	M
M.8	Asignación de frecuencias extendidas.	M	O	O
M.9	Traspaso de portadora intra-celda.	M	C302	C302
M.10	Traspaso de portadora inter-celda.	M	O	O
M.11	Traspaso de conexión, intra-celda.	M	C303	C303
M.12	Traspaso de conexión, inter-celda.	M	O	O
M.13	Soporte de SARI.	M	O	O
M.14	Desactivación de cifrado	C304	C304	C304
C301: Si característica N.17 o N.17, entonces M, en otro caso I. C302: Si servicio M.11, entonces O, en otro caso M. C302: Si servicio M.9, entonces O, en otro caso M. C303: Si servicio N.29 o N.28, entonces M, en otro caso I.				

SPICUM servicio de publicaciones

Apéndice

G

APÉNDICE G: ESTÁNDARES DECT

A continuación se listan las normas publicadas por ETSI para el Sistema DECT. Se han clasificado en función de su contenido y se presentan en primer lugar los documentos más generales, y posteriormente los relacionados con perfiles específicos. En total hay 196 documentos. Para referenciar cada documento se utiliza la nomenclatura posterior a 1998.

Tabla G.1: Visión Global.

Documento	Título
TR 101 178	Digital Enhanced Cordless Telecommunications (DECT); A High Level Guide to the DECT Standardization
TR 102 183	Digital Enhanced Cordless Telecommunications (DECT); Conformance testing on DECT equipment
ETR 056	Digital Enhanced Cordless Telecommunications (DECT); System description document

Tabla G.2: Interfaz Común (CI).

Documento	Título
Especificaciones de Sistema	
TBR 006	Digital Enhanced Cordless Telecommunications (DECT); General terminal attachment requirements
TBR 010	Digital Enhanced Cordless Telecommunications (DECT); General Terminal Attachment Requirements; Telephony Applications
ETR 043	Digital Enhanced Cordless Telecommunications (DECT); Common Interface (CI); Services and facilities requirements specification
EN 300 175-1	Digital Enhanced Cordless Telecommunications (DECT); Common Interface (CI); Part 1: Overview
EN 300 175-2	Digital Enhanced Cordless Telecommunications (DECT); Common Interface (CI); Part 2: Physical Layer (PHL)
EN 300 175-3	Digital Enhanced Cordless Telecommunications (DECT); Common Interface (CI); Part 3: Medium Access Control (MAC) layer
EN 300 175-4	Digital Enhanced Cordless Telecommunications (DECT); Common Interface (CI); Part 4: Data Link Control (DLC) layer
EN 300 175-5	Digital Enhanced Cordless Telecommunications (DECT); Common Interface (CI); Part 5: Network (NWK) layer
EN 300 175-6	Digital Enhanced Cordless Telecommunications (DECT); Common Interface (CI); Part 6: Identities and addressing

SPICUM servicio de publicaciones

Documento	Título
EN 300 175-7	Digital Enhanced Cordless Telecommunications (DECT); Common Interface (CI); Part 7: Security features
EN 300 175-8	Digital Enhanced Cordless Telecommunications (DECT); Common Interface (CI); Part 8: Speech coding and transmission
Especificaciones de Prueba	
I-ETS 300 176	Digital Enhanced Cordless Telecommunications (DECT); Approval test specification
EN 300 176-1	Digital Enhanced Cordless Telecommunications (DECT); Test specification; Part 1: Radio
EN 300 176-2	Digital Enhanced Cordless Telecommunications (DECT); Test specification; Part 2: Speech
EN 300 476-1	Digital Enhanced Cordless Telecommunications (DECT); Common Interface (CI); Protocol Implementation Conformance Statement (PICS) proforma; Part 1: Network (NWK) layer - Portable radio Termination (PT)
EN 300 476-2	Digital Enhanced Cordless Telecommunications (DECT); Common Interface (CI); Protocol Implementation Conformance Statement (PICS) proforma; Part 2: Data Link Control (DLC) layer - Portable radio Termination (PT)
EN 300 476-3	Digital Enhanced Cordless Telecommunications (DECT); Common Interface (CI); Protocol Implementation Conformance Statement (PICS) proforma; Part 3: Medium Access Control (MAC) layer - Portable radio Termination (PT)
EN 300 476-4	Digital Enhanced Cordless Telecommunications (DECT); Common Interface (CI); Protocol Implementation Conformance Statement (PICS) proforma; Part 4: Network (NWK) layer - Fixed radio Termination (FT)
EN 300 476-5	Digital Enhanced Cordless Telecommunications (DECT); Common Interface (CI); Protocol Implementation Conformance Statement (PICS) proforma; Part 5: Data Link Control (DLC) layer - Fixed radio Termination (FT)
EN 300 476-6	Digital Enhanced Cordless Telecommunications (DECT); Common Interface (CI); Protocol Implementation Conformance Statement (PICS) proforma; Part 6: Medium Access Control (MAC) layer - Fixed radio Termination (FT)
EN 300 476-7	Digital Enhanced Cordless Telecommunications (DECT); Common Interface (CI); Protocol Implementation Conformance Statement (PICS) proforma; Part 7: Physical layer
EN 300 497-1	Digital Enhanced Cordless Telecommunications (DECT); Common Interface (CI); Test Case Library (TCL); Part 1: Test Suite Structure (TSS) and Test Purposes (TP) for Medium Access Control (MAC) layer
EN 300 497-2	Digital Enhanced Cordless Telecommunications (DECT); Common Interface (CI); Test Case Library (TCL); Part 2: Abstract Test Suite (ATS) for Medium Access Control (MAC) layer - Portable radio Termination (PT)
EN 300 497-3	Digital Enhanced Cordless Telecommunications (DECT); Common Interface (CI); Test Case Library (TCL); Part 3: Abstract Test Suite (ATS) for Medium Access Control (MAC) layer - Fixed radio Termination (FT)
EN 300 497-4	Digital Enhanced Cordless Telecommunications (DECT); Common Interface (CI); Test Case Library (TCL); Part 4: Test Suite Structure (TSS) and Test Purposes (TP) - Data Link Control (DLC) layer
EN 300 497-5	Digital Enhanced Cordless Telecommunications (DECT); Common Interface (CI); Test Case Library (TCL); Part 5: Abstract Test Suite (ATS) - Data Link Control (DLC) layer
EN 300 497-6	Digital Enhanced Cordless Telecommunications (DECT); Common Interface (CI); Test Case Library (TCL); Part 6: Test Suite Structure (TSS) and Test Purposes (TP) - Network (NWK) layer - Portable radio Termination (PT)
EN 300 497-7	Digital Enhanced Cordless Telecommunications (DECT); Common Interface (CI); Test Case Library (TCL); Part 7: Abstract Test Suite (ATS) for Network (NWK) layer - Portable radio Termination (PT)
EN 300 497-8	Digital Enhanced Cordless Telecommunications (DECT); Common Interface (CI); Test Case Library (TCL); Part 8: Test Suite Structure (TSS) and Test Purposes (TP) - Network (NWK) layer - Fixed radio Termination (FT)
EN 300 497-9	Digital Enhanced Cordless Telecommunications (DECT); Common Interface (CI); Test Case Library (TCL); Part 9: Abstract Test Suite (ATS) for Network (NWK) layer - Fixed radio Termination (FT)

SPICUM servicio de publicaciones

Tabla G.3: Perfil de Acceso Genérico (GAP).

Documento	Título
Especificaciones de Sistema	
EN 300 444	Digital Enhanced Cordless Telecommunications (DECT); Generic Access Profile (GAP)
Especificaciones de Prueba	
TBR 022	Radio Equipment and Systems (RES); Attachment requirements for terminal equipment for Digital Enhanced Cordless Telecommunications (DECT) Generic Access Profile (GAP) applications
TBR 022/A1	Radio Equipment and Systems (RES); Attachment requirements for terminal equipment for Digital Enhanced Cordless Telecommunications (DECT) Generic Access Profile (GAP) applications
EN 300 474-1	Digital Enhanced Cordless Telecommunications (DECT); Generic Access Profile (GAP); Profile requirement list and profile specific Implementation Conformance Statement (ICS) proforma; Part 1: Portable radio Termination (PT)
EN 300 474-2	Digital Enhanced Cordless Telecommunications (DECT); Generic Access Profile (GAP); Profile requirement list and profile specific Implementation Conformance Statement (ICS) proforma; Part 2: Fixed radio Termination (FT)
EN 300 494-1	Digital Enhanced Cordless Telecommunications (DECT); Generic Access Profile (GAP); Profile Test Specification (PTS); Part 1: Summary
EN 300 494-2	Digital Enhanced Cordless Telecommunications (DECT); Generic Access Profile (GAP); Profile Test Specification (PTS); Part 2: Profile Specific Test Specification (PSTS) – Portable radio Termination (PT)
EN 300 494-3	Digital Enhanced Cordless Telecommunications (DECT); Generic Access Profile (GAP); Profile Test Specification (PTS); Part 3: Profile Specific Test Specification (PSTS) – Fixed radio Termination (FT)

Tabla G.4: Perfil de Acceso Público (PAP).

Documento	Título
Especificaciones de Sistema	
ETS 300 175-9[1]	Digital Enhanced Cordless Telecommunications (DECT); Common Interface (CI); Part 9: Public Access Profile (PAP)
TBR 011	Digital Enhanced Cordless Telecommunications (DECT); Attachment requirements for terminal equipment for DECT Public Access Profile (PAP) applications
TBR 011/A1	Digital Enhanced Cordless Telecommunications (DECT); Attachment requirements for terminal equipment for DECT Public Access Profile (PAP) applications
Especificaciones de Prueba	
ETS 300 323-1	Digital Enhanced Cordless Telecommunications (DECT); Public Access Profile (PAP) test specification; Part 1: Overview
ETS 300 323-1 / A1	Digital Enhanced Cordless Telecommunications (DECT); Public Access Profile (PAP) test specification; Part 1: Overview
ETS 300 323-2	Digital Enhanced Cordless Telecommunications (DECT); Public Access Profile (PAP) test specification; Part 2: PT Abstract Test Suite (ATS)
ETS 300 323-3	Digital Enhanced Cordless Telecommunications (DECT); Public Access Profile (PAP) test specification; Part 3: PT PICS proforma
ETS 300 323-3 / A1	Digital Enhanced Cordless Telecommunications (DECT); Public Access Profile (PAP) test specification; Part 3: PT PICS proforma
ETS 300 323-4	Digital Enhanced Cordless Telecommunications (DECT); Public Access Profile (PAP) test specification; Part 4: PT PIXIT proforma
ETS 300 323-5	Digital Enhanced Cordless Telecommunications (DECT); Public Access Profile (PAP) test specification; Part 5: FT Abstract Test Suite (ATS)
ETS 300 323-6	Digital Enhanced Cordless Telecommunications (DECT); Public Access Profile (PAP) test specification; Part 6: FT PICS proforma

[1] *Historical*

SPICUM servicio de publicaciones

Documento	Título
ETS 300 323-6 / A1	Digital Enhanced Cordless Telecommunications (DECT); Public Access Profile (PAP) test specification; Part 6: FT PICS proforma
ETS 300 323-7	Digital Enhanced Cordless Telecommunications (DECT); Public Access Profile (PAP) test specification; Part 7: FT PIXIT proforma

Tabla G.5: Estación Repetidora Inalámbrica (WRS).

Documento	Título
Especificaciones de Sistema	
ETR 246	Digital Enhanced Cordless Telecommunications (DECT); Application of DECT Wireless Relay Stations (WRS)
EN 300 700	Digital Enhanced Cordless Telecommunications (DECT); Wireless Relay Station (WRS)
Especificaciones de Prueba	
TS 101 808-1	Digital Enhanced Cordless Telecommunications (DECT); Wireless Relay Station (WRS); Test Case Library (TCL); Part 1: Test Suite Structure (TSS) and Test Purposes (TP) for Medium Access Control (MAC) layer
TS 101 808-2	Digital Enhanced Cordless Telecommunications (DECT); Wireless Relay Station (WRS); Test Case Library (TCL); Part 2: Abstract Test Suite (ATS) for Medium Access Control (MAC) layer - Cordless Radio Fixed Part Portable radio Termination (CRFP_PT)
TS 101 808-3	Digital Enhanced Cordless Telecommunications (DECT); Wireless Relay Station (WRS); Test Case Library (TCL); Part 3: Abstract Test Suite (ATS) for Medium Access Control (MAC) layer - Cordless Radio Fixed Part Fixed radio Termination (CRFP_FT)
TS 101 808-4	Digital Enhanced Cordless Telecommunications (DECT); Wireless Relay Station (WRS); Test Case Library (TCL); Part 4: Test Suite Structure (TSS) and Test Purposes (TP) - Data Link Control (DLC) layer
TS 101 808-5	Digital Enhanced Cordless Telecommunications (DECT); Wireless Relay Station (WRS); Test Case Library (TCL); Part 5: Abstract Test Suite (ATS) - Data Link Control (DLC) layer;
TS 101 808-6	Digital Enhanced Cordless Telecommunications (DECT); Wireless Relay Station (WRS); Test Case Library (TCL); Part 6: Abstract Test Suite (ATS) - Data Link Control (DLC) layer;
TS 101 808-7	Digital Enhanced Cordless Telecommunications (DECT); Wireless Relay Station (WRS); Test Case Library (TCL); Part 7: Test Suite Structure (TSS) and Test Purposes (TP) - Network (NWK) layer
TS 101 808-8	Digital Enhanced Cordless Telecommunications (DECT); Wireless Relay Station (WRS); Test Case Library (TCL); Part 8: Abstract Test Suite (ATS) for Network (NWK) layer - Cordless Radio Fixed Part Portable radio Termination (CRFP_PT)
TS 101 808-9	Digital Enhanced Cordless Telecommunications (DECT); Wireless Relay Station (WRS); Test Case Library (TCL); Part 9: Abstract Test Suite (ATS) for Network (NWK) layer - Cordless Radio Fixed Part Fixed radio Termination (CRFP_FT)

Tabla G.6: Módulo de Autentificación DECT (DAM).

Documento	Título
Especificaciones de Sistema	
ETS 300 331	Digital Enhanced Cordless Telecommunications (DECT); DECT Authentication Module (DAM);
ETS 300 825	Digital Enhanced Cordless Telecommunications (DECT); 3 Volt DECT Authentication Module (DAM)
Especificaciones de Prueba	
ETS 300 759	Digital Enhanced Cordless Telecommunications (DECT); DECT Authentication Module (DAM); Test specification for DAM
ETS 300 760	Digital Enhanced Cordless Telecommunications (DECT); DECT Authentication Module (DAM); Implementation Conformance Statement (ICS) proforma specification

SPICUM servicio de publicaciones

Tabla G.7: Movilidad de Terminales Inalámbricos (CTM).

Documento	Título
Especificaciones de Sistema	
EN 300 824	Digital Enhanced Cordless Telecommunications (DECT); Cordless Terminal Mobility (CTM); CTM Access Profile (CAP)
EN 301 361-1	Digital Enhanced Cordless Telecommunications (DECT); Integrated Services Digital Network (ISDN); ISDN Mobility protocol Interworking specification Profile (IMIP); Part 1: DECT/ISDN interworking for Cordless Terminal Mobility (CTM) support
EN 302 096[1]	Digital Enhanced Cordless Telecommunications (DECT); Cordless Terminal Mobility (CTM); Feature Package 1 (FP1); CTM circuit-switched data profile, 32 kbit/s and 64 kbit/s Unrestricted Digital Information (UDI)
Especificaciones de Prueba	
EN 301 371-1	Digital Enhanced Cordless Telecommunications (DECT); Cordless Terminal Mobility (CTM); CTM Access Profile (CAP); Profile Test Specification (PTS);
EN 301 371-2	Digital Enhanced Cordless Telecommunications (DECT); Cordless Terminal Mobility (CTM); CTM Access Profile (CAP); Profile Test Specification (PTS);
EN 301 371-3	Digital Enhanced Cordless Telecommunications (DECT); Cordless Terminal Mobility (CTM); CTM Access Profile (CAP); Profile Test Specification (PTS);

Tabla G.8: Bucle Local Radio (RLL).

Documento	Título
ETR 308	Digital Enhanced Cordless Telecommunications (DECT); Services, facilities and configurations for DECT in the local loop
EN 300 765-1	Digital Enhanced Cordless Telecommunications (DECT); Radio in the Local Loop (RLL) Access Profile (RAP); Part 1: Basic telephony services
EN 300 765-2	Digital Enhanced Cordless Telecommunications (DECT); Radio in the Local Loop (RLL) Access Profile (RAP); Part 2: Advanced telephony services

Tabla G.9: Interconexión con GSM.

Documento	Título
Especificaciones de Sistema	
TBR 036	Digital Enhanced Cordless Telecommunications (DECT); Global System for Mobile communications (GSM); DECT access to GSM Public Land Mobile Networks (PLMNs) for 3,1 kHz speech applications
ETR 159	Digital Enhanced Cordless Telecommunications (DECT); Global System for Mobile communications (GSM); Wide area mobility using GSM
ETR 341	Digital Enhanced Cordless Telecommunications (DECT); Global System for Mobile communications (GSM); DECT/GSM Interworking Profile (IWP); Profile overview
EN 300 370	Digital Enhanced Cordless Telecommunications (DECT); Global System for Mobile communications (GSM); DECT/GSM Interworking Profile (IWP); Access and mapping (protocol/procedure description for 3,1 kHz speech service)
EN 300 466	Digital Enhanced Cordless Telecommunications (DECT); Global System for Mobile communications (GSM); DECT/GSM Interworking Profile (IWP); General description of service requirements;
ETS 300 499	Digital Enhanced Cordless Telecommunications (DECT); Global System for Mobile communications (GSM); DECT/GSM Interworking Profile (IWP); Mobile services Switching Centre (MSC) - Fixed Part (FP) interconnection
EN 300 703	Digital Enhanced Cordless Telecommunications (DECT); Global System for Mobile communications (GSM); DECT/GSM Interworking Profile (IWP); GSM Phase 2 supplementary services implementation
ETS 300 756	Digital Enhanced Cordless Telecommunications (DECT); Global System for Mobile communications (GSM); DECT/GSM Interworking Profile (IWP); Implementation of bearer services

SPICUM servicio de publicaciones

Documento	Título
ETS 300 764	Digital Enhanced Cordless Telecommunications (DECT); Global System for Mobile communications (GSM); DECT/GSM Interworking Profile (IWP); Implementation of short message service, point-to-point and cell broadcast
ETS 300 787	Digital Enhanced Cordless Telecommunications (DECT); Global System for Mobile communications (GSM); Integrated Services Digital Network (ISDN); DECT access to GSM via ISDN
ETS 300 788	Digital Enhanced Cordless Telecommunications (DECT); Global System for Mobile communications (GSM); Integrated Services Digital Network (ISDN); DECT access to GSM via ISDN
ETS 300 792	Digital Enhanced Cordless Telecommunications (DECT); Global System for Mobile communications (GSM); DECT/GSM Interworking Profile (IWP); Implementation of facsimile group 3
EN 301 361-2	Digital Enhanced Cordless Telecommunications (DECT); Integrated Services Digital Network (ISDN); ISDN Mobility protocol Interworking specification Profile (IMIP); Part 2: DECT/ISDN interworking for Global System for Mobile communications (GSM) support
TS 101 623	Digital cellular telecommunications system (Phase 2+) (GSM); ISDN-based DECT/GSM interworking; Feasibility study (GSM 01.48 version 6.0.1 Release 1997)
GSM 01.48	Digital cellular telecommunications system (Phase 2+) (GSM); ISDN-based DECT/GSM interworking; Feasibility study (GSM 01.48)
Especificaciones de Prueba	
ETS 300 702-1	Digital Enhanced Cordless Telecommunications (DECT); Global System for Mobile communications (GSM); DECT/GSM Interworking Profile (IWP); Part 1: Profile Test Specification (PTS) summary
ETS 300 702-2	Digital Enhanced Cordless Telecommunications (DECT); Global System for Mobile communications (GSM); DECT/GSM Interworking Profile (IWP); Profile Test Specification (PTS);
ETS 300 702-3	Digital Enhanced Cordless Telecommunications (DECT); Global System for Mobile communications (GSM); DECT/GSM Interworking Profile (IWP); Profile Test Specification (PTS);
ETS 300 704-1	Digital Enhanced Cordless Telecommunications (DECT); Global System for Mobile communications (GSM); DECT/GSM Interworking Profile (IWP); Profile Implementation Conformance Statement (ICS);
ETS 300 704-2	Digital Enhanced Cordless Telecommunications (DECT); Global System for Mobile communications (GSM); DECT/GSM Interworking Profile (IWP); Profile Implementation Conformance Statement (ICS);

Tabla G.10: Interconexión con ISDN.

Documento	Título
Especificaciones de Sistema	
TBR 040	Digital Enhanced Cordless Telecommunications (DECT); Integrated Services Digital Network (ISDN); Attachment requirements for terminal equipment for DECT/ISDN interworking profile applications
EN 300 434-1	Digital Enhanced Cordless Telecommunications (DECT); Integrated Services Digital Network (ISDN); DECT/ISDN interworking for end system configuration; Part 1: Interworking specification
EN 300 434-2	Digital Enhanced Cordless Telecommunications (DECT); Integrated Services Digital Network (ISDN); DECT/ISDN interworking for end system configuration; Part 2: Access profile
ETS 300 787	Digital Enhanced Cordless Telecommunications (DECT); Global System for Mobile communications (GSM); Integrated Services Digital Network (ISDN); DECT access to GSM via ISDN;
ETS 300 788	Digital Enhanced Cordless Telecommunications (DECT); Global System for Mobile communications (GSM); Integrated Services Digital Network (ISDN); DECT access to GSM via ISDN;
EN 300 822	Digital Enhanced Cordless Telecommunications (DECT); Integrated Services Digital Network (ISDN); DECT/ISDN interworking for intermediate system configuration; Interworking and profile specification

SPICUM servicio de publicaciones

Documento	Título
EN 301 361-1	Digital Enhanced Cordless Telecommunications (DECT); Integrated Services Digital Network (ISDN); ISDN Mobility protocol Interworking specification Profile (IMIP); Part 1: DECT/ISDN interworking for Cordless Terminal Mobility (CTM) support
EN 301 361-2	Digital Enhanced Cordless Telecommunications (DECT); Integrated Services Digital Network (ISDN); ISDN Mobility protocol Interworking specification Profile (IMIP); Part 2: DECT/ISDN interworking for Global System for Mobile communications (GSM) support
EN 301 440	Digital Enhanced Cordless Telecommunications (DECT); Integrated Services Digital Network (ISDN); Attachment requirements for terminal equipment for DECT/ISDN interworking profile applications
TS 101 623	Digital cellular telecommunications system (Phase 2+) (GSM); ISDN-based DECT/GSM interworking; Feasibility study (GSM 01.48 version 6.0.1 Release 1997)
TS 101 679	Digital Enhanced Cordless Telecommunications (DECT); Broadband Integrated Services Digital Network (B-ISDN); DECT/B-ISDN interworking
GSM 01.48	Digital cellular telecommunications system (Phase 2+) (GSM); ISDN-based DECT/GSM interworking; Feasibility study (GSM 01.48)
Especificaciones de Prueba	
ETS 300 705-1	Digital Enhanced Cordless Telecommunications (DECT); Integrated Services Digital Network (ISDN); DECT/ISDN interworking for end system configuration; Profile Implementation Conformance Statement (ICS);
ETS 300 705-2	Digital Enhanced Cordless Telecommunications (DECT); Integrated Services Digital Network (ISDN); DECT/ISDN interworking for end system configuration; Profile Implementation Conformance Statement (ICS);
ETS 300 758-1	Digital Enhanced Cordless Telecommunications (DECT); Integrated Services Digital Network (ISDN); DECT/ISDN interworking for end system configuration; Profile Test Specification (PTS);
ETS 300 758-2	Digital Enhanced Cordless Telecommunications (DECT); Integrated Services Digital Network (ISDN); DECT/ISDN interworking for end system configuration; Profile Test Specification (PTS);
ETS 300 758-3	Digital Enhanced Cordless Telecommunications (DECT); Integrated Services Digital Network (ISDN); DECT/ISDN interworking for end system configuration; Profile Test Specification (PTS);
EN 301 241-1	Digital Enhanced Cordless Telecommunications (DECT); Integrated Services Digital Network (ISDN); DECT/ISDN interworking for intermediate system configuration; Profile Implementation Conformance Statement (ICS);
EN 301 241-2	Digital Enhanced Cordless Telecommunications (DECT); Integrated Services Digital Network (ISDN); DECT/ISDN interworking for intermediate system configuration; Profile Implementation Conformance Statement (ICS);
EN 301 614-1	Digital Enhanced Cordless Telecommunications (DECT); Integrated Services Digital Network (ISDN); DECT/ISDN interworking for intermediate system configuration; Part 1: Profile Test Specification (PTS) summary
EN 301 614-2	Digital Enhanced Cordless Telecommunications (DECT); Integrated Services Digital Network (ISDN); DECT/ISDN interworking for intermediate system configuration; Part 2: Profile Specific Test Specification (PSTS) for Portable radio Termination (PT)
EN 301 614-3	Digital Enhanced Cordless Telecommunications (DECT); Integrated Services Digital Network (ISDN); DECT/ISDN interworking for intermediate system configuration; Part 3: Profile Specific Test Specification (PSTS) for Fixed radio Termination (FT)

Tabla G.11: Terminales Duales.

Documento	Título
TR 101 072	Digital Enhanced Cordless Telecommunications (DECT); Global System for Mobile communications (GSM); DECT/GSM integration based on dual-mode terminals
TR 101 176	Digital Enhanced Cordless Telecommunications (DECT); Global System for Mobile communications (GSM); DECT/GSM advanced integration of DECT/GSM dual-mode terminal equipment
EN 301 242	Digital Enhanced Cordless Telecommunications (DECT); Global System for Mobile communications (GSM); DECT/GSM integration based on dual-mode terminals
EN 301 439	Digital Enhanced Cordless Telecommunications (DECT); Global System for Mobile

SPICUM servicio de publicaciones

Documento	Título
	communications (GSM); Attachment requirements for DECT/GSM dual-mode terminal equipment

Tabla G.12: Interconexión con UMTS.

Documento	Título
TS 101 863-1	Digital Enhanced Cordless Telecommunications (DECT); DECT/UMTS Interworking Profile (IWP); Part 1: General description and overview
TS 101 863-2	Digital Enhanced Cordless Telecommunications (DECT); DECT/UMTS Interworking Profile (IWP); Part 2: CN-FP interworking
TS 101 863-3	Digital Enhanced Cordless Telecommunications (DECT); DECT/UMTS Interworking Profile (IWP); Part 3: 3,1 kHz speech service
TS 101 863-4	Digital Enhanced Cordless Telecommunications (DECT); DECT/UMTS Interworking Profile (IWP); Part 4: Supplementary services
TS 101 863-5	Digital Enhanced Cordless Telecommunications (DECT); DECT/UMTS Interworking Profile (IWP); Part 5: SMS point-to-point and cell broadcast
TS 101 863-6	Digital Enhanced Cordless Telecommunications (DECT); DECT/UMTS Interworking Profile (IWP); Part 6: Packet switched data

Tabla G.13: Interconexión con Redes IP.

Documento	Título
TR 102 010	Digital Enhanced Cordless Telecommunications (DECT); DECT access to IP networks;
TS 102 265	Digital Enhanced Cordless Telecommunications (DECT); DECT access to IP networks

Tabla G.14: Perfiles de Servicios de Datos (DSP).

Documento	Título
Especificaciones de Sistema	
TR 102 185	Digital Enhanced Cordless Telecommunications (DECT); Data Services Profile (DSP); Profile overview
TR 102 179	Digital Enhanced Cordless Telecommunications (DECT); AT command interface; High-level description;
ETS 300 435[1]	Digital Enhanced Cordless Telecommunications (DECT); Data Services Profile (DSP); Base standard including interworking to connectionless networks (service types A and B, class 1)
ETS 300 651[1]	Digital Enhanced Cordless Telecommunications (DECT); Data Services Profile (DSP); Generic data link service (service type C, class 2)
ETS 300 699[1]	Digital Enhanced Cordless Telecommunications (DECT); Data Services Profile (DSP); Generic data link service for closed user groups (service type C, class 1)
ETS 300 701[1]	Digital Enhanced Cordless Telecommunications (DECT); Data Services Profile (DSP); Generic frame relay service with mobility (service types A and B, class 2)
ETS 300 755[1]	Digital Enhanced Cordless Telecommunications (DECT); Data Services Profile (DSP); Multimedia Messaging Service (MMS) with specific provision for facsimile services (service type F, class 2)
EN 301 240[1]	Digital Enhanced Cordless Telecommunications (DECT); Data Services Profile (DSP); Point-to-Point Protocol (PPP) interworking for internet access and general multi-protocol datagram transport

Tabla G.15: Servicio de Paquetes Radio DECT (DPRS).

Documento	Título
Especificaciones de Sistema	
EN 301 649	Digital Enhanced Cordless Telecommunications (DECT); DECT Packet Radio Service (DPRS)
Especificaciones de Prueba	

SPICUM servicio de publicaciones

Documento	Título
EN 301 469-1	Digital Enhanced Cordless Telecommunications (DECT); DECT Packet Radio Service (DPRS) Test Case Library (TCL); Part 1: Test Suite Structure (TSS) and Test Purposes (TP) - Medium Access Control (MAC) layer
EN 301 469-2	Digital Enhanced Cordless Telecommunications (DECT); DECT Packet Radio Service (DPRS) Test Case Library (TCL); Part 2: Abstract Test Suite (ATS) - Medium Access Control (MAC) layer - Portable radio Termination (PT)
EN 301 469-3	Digital Enhanced Cordless Telecommunications (DECT); DECT Packet Radio Service (DPRS) Test Case Library (TCL); Part 3: Abstract Test Suite (ATS) - Medium Access Control (MAC) layer - Fixed radio Termination (FT)
EN 301 469-4	Digital Enhanced Cordless Telecommunications (DECT); DECT Packet Radio Service (DPRS) Test Case Library (TCL); Part 4: Test Suite Structure (TSS) and Test Purposes (TP) - Data Link Control (DLC) layer
EN 301 469-5	Digital Enhanced Cordless Telecommunications (DECT); DECT Packet Radio Service (DPRS) Test Case Library (TCL); Part 5: Abstract Test Suite (ATS) - Data Link Control (DLC) layer - Portable radio Termination (PT)
EN 301 469-6	Digital Enhanced Cordless Telecommunications (DECT); DECT Packet Radio Service (DPRS) Test Case Library (TCL); Part 6: Abstract Test Suite (ATS) - Data Link Control (DLC) layer - Fixed radio Termination (FT)
EN 301 469-7	Digital Enhanced Cordless Telecommunications (DECT); DECT Packet Radio Service (DPRS) Test Case Library (TCL); Part 7: Test Suite Structure (TSS) and Test Purposes (TP) - Network (NWK) layer
EN 301 469-8	Digital Enhanced Cordless Telecommunications (DECT); DECT Packet Radio Service (DPRS) Test Case Library (TCL); Part 8: Abstract Test Suite (ATS) - Network (NWK) layer - Portable radio Termination (PT)
EN 301 469-9	Digital Enhanced Cordless Telecommunications (DECT); DECT Packet Radio Service (DPRS) Test Case Library (TCL); Part 9: Abstract Test Suite (ATS) - Network (NWK) layer - Fixed radio Termination (FT)
TS 101 869-1	Digital Enhanced Cordless Telecommunications (DECT); DECT Packet Radio Services (DPRS); Profile requirement list and profile specific Implementation Conformance Statement (ICS) proforma; Part 1: Portable radio Termination (PT)
TS 101 869-2	Digital Enhanced Cordless Telecommunications (DECT); DECT Packet Radio Services (DPRS); Profile requirement list and profile specific Implementation Conformance Statement (ICS) proforma; Part 2: Fixed radio Termination (FT)
TS 101 950	Digital Enhanced Cordless Telecommunications (DECT); DECT Packet Radio Service (DPRS); Interoperability Test Specification

Tabla G.16: Perfil de Acceso Multimedia DECT (DMAP).

Documento	Título
Especificaciones de Sistema	
EN 301 650	Digital Enhanced Cordless Telecommunications (DECT); DECT Multimedia Access Profile (DMAP); Application Specific Access Profile (ASAP)
Especificaciones de Prueba	
TS 101 859-1	Digital Enhanced Cordless Telecommunications (DECT); DECT Multimedia Access Profile (DMAP); Application Specific Access Profile (ASAP); Profile Test Specification (PTS)
TS 101 859-2	Digital Enhanced Cordless Telecommunications (DECT); DECT Multimedia Access Profile (DMAP); Application Specific Access Profile (ASAP); Profile Test Specification (PTS)
TS 101 859-3	Digital Enhanced Cordless Telecommunications (DECT); DECT Multimedia Access Profile (DMAP); Application Specific Access Profile (ASAP); Profile Test Specification (PTS)
TS 101 871-1	Digital Enhanced Cordless Telecommunications (DECT); Application Specific Access Profile (ASAP); DECT Multimedia Access Profile (DMAP); Profile requirement list and profile specific Implementation Conformance Statement (ICS) proforma
TS 101 871-2	Digital Enhanced Cordless Telecommunications (DECT); Application Specific Access Profile (ASAP); DECT Multimedia Access Profile (DMAP); Profile requirement list and profile specific Implementation Conformance Statement (ICS) proforma

SPICUM servicio de publicaciones

Tabla G.17: Interconexión con Ethernet.

Documento	Título
Especificaciones de Sistema	
TS 101 942	Digital Enhanced Cordless Telecommunications (DECT); DECT Packet Radio Service (DPRS); Application Specific Access Profile (ASAP): Ethernet (Eth) Interworking
Especificaciones de Prueba	
TS 102 013-1	Digital Enhanced Cordless Telecommunications (DECT); DECT Packet Radio Service (DPRS); Application Specific Access Profile (ASAP): Ethernet Interworking; Profile Implementation Conformance Statement (ICS)
TS 102 013-2	Digital Enhanced Cordless Telecommunications (DECT); DECT Packet Radio Service (DPRS); Application Specific Access Profile (ASAP): Ethernet Interworking; Profile Implementation Conformance Statement (ICS)
TS 102 014	Digital Enhanced Cordless Telecommunications (DECT); DECT Packet Radio Service (DPRS); Application Specific Access Profile (ASAP): Ethernet Interworking; Profile Test Specification (PTS)

Tabla G.18: Interconexión con V.24.

Documento	Título
Especificaciones de Sistema	
TS 101 947	Digital Enhanced Cordless Telecommunications (DECT); DECT Packet Radio Service (DPRS); Application Specific Access Profile (ASAP): V.24 Interworking
Especificaciones de Prueba	
TS 102 011-1	Digital Enhanced Cordless Telecommunications (DECT); DECT Packet Radio Service (DPRS); Application Specific Access Profile (ASAP): V.24 Interworking; Profile Implementation Conformance Statement (ICS)
TS 102 011-2	Digital Enhanced Cordless Telecommunications (DECT); DECT Packet Radio Service (DPRS); Application Specific Access Profile (ASAP): V.24 Interworking; Profile Implementation Conformance Statement (ICS)
TS 102 012	Digital Enhanced Cordless Telecommunications (DECT); DECT Packet Radio Service (DPRS); Application Specific Access Profile (ASAP): V.24 Interworking; Profile Test Specification (PTS)

Tabla G.19: Perfiles D.

Documento	Título
Especificaciones de Sistema	
EN 301 238	Digital Enhanced Cordless Telecommunications (DECT); Data Services Profile (DSP); Isochronous data bearer services with roaming mobility (service type D, mobility class 2)
EN 301 239	Digital Enhanced Cordless Telecommunications (DECT); Data Services Profile (DSP); Isochronous data bearer services for closed user groups (service type D, mobility class 1)
Especificaciones de Prueba	
TS 101 945-1	Digital Enhanced Cordless Telecommunications (DECT); Data Services Profile (DSP); Isochronous data bearer services with roaming capability (Service Type D, mobility class 2); Profile requirement list and profile specific Implementation Conformance Statement (ICS) proforma
TS 101 945-2	Digital Enhanced Cordless Telecommunications (DECT); Data Services Profile (DSP); Isochronous data bearer services with roaming capability (Service Type D, mobility class 2); Profile requirement list and profile specific Implementation Conformance Statement (ICS) proforma

Tabla G.20: Perfil Abierto de Acceso a Datos (ODAP).

Documento	Título
TS 102 342	Digital Enhanced Cordless Telecommunications (DECT); Cordless multimedia communication system; Open Data Access Profile (ODAP)

SPICUM servicio de publicaciones

Tabla G.21: Servicios de Mensajes a Baja Velocidad (LRMS).

Documento	Título
Especificaciones de Sistema	
EN 300 757	Digital Enhanced Cordless Telecommunications (DECT); Low Rate Messaging Service (LRMS) including Short Messaging Service (SMS)
Especificaciones de Prueba	
TS 101 946-1	Digital Enhanced Cordless Telecommunications (DECT); Low Rate Messaging Service (LRMS) including Short Message Service (SMS); Profile requirement list and profile specific Implementation Conformance Statement (ICS) proforma; Part 1: Portable radio Termination (PT)
TS 101 946-2	Digital Enhanced Cordless Telecommunications (DECT); Low Rate Messaging Service (LMRS) including Short Message Service (SMS); Profile requirement list and profile specific Implementation Conformance Statement (ICS) proforma; Part 2: Fixed radio Termination (FT)

Tabla G.22: Servicio de Mensajes Multimedia por Red Fija (F-MMS).

Documento	Título
TS 102 379	Digital Enhanced Cordless Telecommunications (DECT); Fixed network Multimedia Message Service (F-MMS) Interworking Profile

Tabla G.23: Difusión de Audio y Video Digital.

Documento	Título
EN 301 193	Digital Video Broadcasting (DVB); Interaction channel through the Digital Enhanced Cordless Telecommunications (DECT)
ES 201 737	Digital Audio Broadcasting (DAB); Interaction channel through Global System for Mobile communications (GSM) the Public switched Telecommunications System (PSTN); Integrated Services Digital Network (ISDN) and Digital Enhanced Cordless Telecommunications (DECT)
TS 101 737	Digital Audio Broadcasting (DAB); Interaction channel through Global System for Mobile communications (GSM) the Public switched Telecommunications System (PSTN); Integrated Services Digital Network (ISDN) and Digital Enhanced Cordless Telecommunications (DECT)

Tabla G.24: DECT de Nueva Generación.

Documento	Título
TR 102 570	Digital Enhanced Cordless Telecommunications (DECT);New Generation DECT;Overview and Requirements
TS 102 527-2	Digital Enhanced Cordless Telecommunications (DECT);New Generation DECT;Part 2: Support of transparent IP packet data
TS 102 527-3	Digital Enhanced Cordless Telecommunications (DECT);New Generation DECT;Part 3: Extended wideband speech services
TS 102 527-1	Digital Enhanced Cordless Telecommunications (DECT);New Generation DECT;Part 1: Wideband speech

Tabla G.25: Compatibilidad Electromagnética.

Documento	Título
ETS 300 329	Radio Equipment and Systems (RES); ElectroMagnetic Compatibility (EMC) for Digital Enhanced Cordless Telecommunications (DECT) equipment
EN 301 406	Digital Enhanced Cordless Telecommunications (DECT); Harmonized EN for Digital Enhanced Cordless Telecommunications (DECT) covering essential requirements under article 3.2 of the R&TTE Directive; Generic radio

SPICUM servicio de publicaciones

Documento	Título
EN 301 489-6	Electromagnetic compatibility and Radio spectrum Matters (ERM); ElectroMagnetic Compatibility (EMC) standard for radio equipment and services; Part 6: Specific conditions for Digital Enhanced Cordless Telecommunications (DECT) equipment
EN 301 908-10	Electromagnetic compatibility and Radio spectrum Matters (ERM); Base Stations (BS), Repeaters and User Equipment (UE) for IMT-2000 Third-Generation cellular networks; Part 10: Harmonized EN for IMT-2000, FDMA/TDMA (DECT) covering essential requirements of article 3.2 of the R&TTE Directive

Tabla G.26: Utilización del Espectro.

Documento	Título
TR 101 159	Digital Enhanced Cordless Telecommunications (DECT); Implementing DECT in an arbitrary spectrum allocation
TR 101 310	Digital Enhanced Cordless Telecommunications (DECT); Traffic capacity and spectrum requirements for multi-system and multi-service DECT applications co-existing in a common frequency band
TR 101 370	Digital Enhanced Cordless Telecommunications (DECT); Implementing DECT Fixed Wireless Access (FWA) in an arbitrary spectrum allocation
ETR 042[1]	Digital Enhanced Cordless Telecommunications (DECT); A guide to DECT features that influence the traffic capacity and the maintenance of high radio link transmission quality, including the results of simulations
ETR 310	Digital Enhanced Cordless Telecommunications (DECT); Traffic capacity and spectrum requirements for multi-system and multi-service DECT applications co-existing in a common frequency band
ETR 310/C1	Digital Enhanced Cordless Telecommunications (DECT); Traffic capacity and spectrum requirements for multi-system and multi-service DECT applications co-existing in a common frequency band

Tabla G.27: Otros Documentos.

Documento	Título
ETR 041	Transmission and Multiplexing (TM); Digital European Cordless Telecommunications (DECT); Transmission aspects 3,1 kHz telephony Interworking with other networks
ETR 015[1]	Digital Enhanced Cordless Telecommunications (DECT); Reference document
TS 101 948[1]	Digital Enhanced Cordless Telecommunications (DECT); DECT derivative for implementation in the 2,45 GHz ISM Band (DECT-ISM)

SPICUM servicio de publicaciones

Apéndice

H

APÉNDICE H: LISTA DE PRUEBAS DECT

Este Apéndice incluye la lista de Grupos de Pruebas y Casos de Prueba de las Pruebas de Conformidad DECT para los niveles DLC y NWK-PT, indicando cuáles son GAP. Para cada Grupo o Caso de Prueba se indica la funcionalidad que verifica. En las tablas solamente se incluyen aquellos Grupos de Pruebas y Casos de Prueba incluidos en la versión final de los Juegos de Pruebas, no teniendo en cuenta aquellos Grupos de Pruebas para los cuales no hay ningún Caso de Prueba definido (ej: para el nivel DLC el Grupo de Pruebas `/U-Plane/Class0/BI/`), ni aquellos Casos de Prueba que han sido identificados como no comprobables (ej: para el nivel NWK de la Terminación Portátil el Caso de Prueba `/PT/MM/BV/PR-01`)[1].

La Tabla H.1 muestra el volumen de Casos de Prueba de cada Nivel, una vez descontados los Casos de Prueba con Propósitos de Prueba no comprobables y los Grupos de Pruebas vacíos. Se han incluido también en la tabla los Casos de Prueba del Nivel MAC y del Nivel NWK_FT por completitud. Hay un total de 300 Casos de Prueba aplicables al perfil GAP; de éstos, 166 son aplicables a los Niveles DLC o NWK-PT.

Tabla H.1: Resumen de los Juegos de Pruebas de Protocolo para DECT.

Nivel	Documento CI	Terminación	CI		GAP	
			Grupos de Pruebas	Casos de Prueba	Grupos de Pruebas	Casos de Prueba
MAC	[EN 300 497-2]	PT	18	27	18	21
	[EN 300 497-3]	FT	19	28	19	23
DLC	[EN 300 497-5]	PT	18	56	11	30
	[EN 300 497-5]	FT	18	56	10	22
NWK	[EN 300 497-7]	PT	33	172	31	114
	[EN 300 497-9]	FT	34	114	30	90

[1] La traducción al español de estas tablas ha sido obtenida de [SALM98].

SPICUM servicio de publicaciones

H.1 Lista de Pruebas para el Nivel DLC

La Tabla H.2 muestra la lista de Grupos de Pruebas [EN 300 497-4] para el nivel DLC y la Tabla H.3 enumera los Casos de Prueba [EN 300 497-5]. La columna GAP indica con 'X' que el Grupo o Caso de Prueba correspondiente es aplicable al perfil GAP ([ETS 300 494-2], [ETS 300 494-3]), diferenciando entre la Terminación Fija y la Terminación Portátil. Un '*' indica que, aunque siendo aplicable al perfil GAP, no se ha podido implementar por limitaciones del Módulo de Capa Física; el número a su lado indica alguna de las siguientes limitaciones:

(1) No poder provocar un traspaso en las placas.

Tabla H.2: Lista de Grupos de Pruebas del Nivel DLC de DECT.

Grupo de Pruebas	Expresión de Selección	Objetivo	GAP	
			FT	PT
C_Plane/	Obligatorio	Conformidad de los comportamientos genéricos del Plano C.	X	X
C_Plane/ClassU/	ClassU_mandatory	Conformidad de los comportamientos de Clase U del Plano C.	---	---
C_Plane/ClassU/CA/	ClassU_mandatory	Conformidad de los comportamientos de capacidad de Clase U del Plano C.	---	---
C_Plane/ClassU/BI/	ClassU_mandatory	Conformidad de los comportamientos inválidos de Clase U del Plano C.	---	---
C_Plane/ClassA/	ClassA_mandatory	Conformidad de los comportamientos de Clase A del Plano C.	X	X
C_Plane/ClassA/CA/	ClassA_mandatory	Conformidad de los comportamientos de capacidad de Clase A del Plano C.	X	X
C_Plane/ClassA/BV/	ClassA_mandatory	Conformidad de los comportamientos válidos de Clase A del Plano C.	X	X
C_Plane/ClassA/BI/	ClassA_mandatory	Conformidad de los comportamientos inválidos de Clase A del Plano C.	X	X
C_Plane/ClassA/BO/	ClassA_mandatory	Conformidad de los comportamientos inoportunos de Clase A del Plano C.	---	X
C_Plane/Lb/	Lb_mandatory	Conformidad de los comportamientos de difusión del Plano C.	X	X
C_Plane/Lb/CA/	Lb_mandatory	Conformidad de los comportamientos de capacidad de difusión del Plano C.	X	X
U_Plane/	Obligatorio	Conformidad de los comportamientos genéricos del Plano U.	X	X
U_Plane/Class0/	Class0_mandatory	Conformidad de los comportamientos de Clase 0 del Plano U.	X	X
U_Plane/Class0/CA/	Class0_mandatory	Conformidad de los comportamientos de capacidad de Clase 0 del Plano U.	X	X
U_Plane/Class1/	Class1_mandatory	Conformidad de los comportamientos de Clase 1 del Plano U.	---	---
U_Plane/Class1/CA/	Class1_mandatory	Conformidad de los comportamientos de capacidad de Clase 1 del Plano U.	---	---
U_Plane/Class1/BV/	Class1_mandatory	Conformidad de los comportamientos válidos de Clase 1 del Plano U.	---	---
U_Plane/Class1/BI/	Class1_mandatory	Conformidad de los comportamientos inválidos de Clase 1 del Plano U.	---	---

Tabla H.3: Lista de Casos de Prueba del Nivel DLC de DECT.

Grupo de Pruebas	Caso de Prueba	Expresión de Selección	Descripción	GAP	
				FT	PT
C_Plane/ClassU/CA/	TC_U_CA_000	ClassU_snd	Verificar que la IUT puede generar una trama UI usando un servicio MAC no orientado a conexión.	---	---
C_Plane/ClassU/CA/	TC_U_CA_002	ClassU_rec	Verificar que la IUT puede recibir una trama UI usando un servicio MAC no orientado a conexión.	---	---
C_Plane/ClassU/CA/	TC_U_CA_003	ClassU_rec_on_co	Verificar que la IUT puede recibir una trama UI usando una conexión MAC abierta.	---	---
C_Plane/ClassU/BI/	TC_U_BI_000	ClassU_rec	Verificar que la IUT, al recibir una trama UI con el bit P puesto a '1', acepta esta trama errónea. La trama UI se transmite en un servicio MAC no orientado a conexión.	---	---
C_Plane/ClassU/BI/	TC_U_BI_001	ClassU_rec_on_co	Verificar que la IUT, al recibir una trama UI con el bit P puesto a '1', acepta esta trama errónea. La trama UI se transmite en una conexión MAC abierta.	---	---
C_Plane/ClassU/BI/	TC_U_BI_002	ClassU_rec	Verificar que la IUT, al recibir una trama UI con el bit NLF puesto a '1', acepta esta trama errónea. La trama UI se transmite en un servicio MAC no orientado a conexión.	---	---
C_Plane/ClassU/BI/	TC_U_BI_003	ClassU_rec_on_co	Verificar que la IUT, al recibir una trama UI con el bit NLF puesto a '1', acepta esta trama errónea. La trama UI se transmite en una conexión MAC abierta.	---	---
C_Plane/ClassU/BI/	TC_U_BI_004	ClassU_rec	Verificar que la IUT descarta una trama UI con un valor LLN inadecuado (no una operación de Clase U). La trama UI se transmite en un servicio MAC no orientado a conexión.	---	---
C_Plane/ClassU/BI/	TC_U_BI_005	ClassU_rec_on_co	Verificar que la IUT descarta una trama UI con un campo LLN inadecuado (no una operación de Clase U). La trama UI se transmite en una conexión MAC abierta.	---	---
C_Plane/ClassU/BI/	TC_U_BI_006	ClassU_rec	Verificar que la IUT descarta una trama UI con un campo SAPI inadecuado (no orientado a conexión). La trama UI se transmite en un servicio MAC no orientado a conexión.	---	---
C_Plane/ClassU/BI/	TC_U_BI_007	ClassU_rec_on_co	Verificar que la IUT descarta una trama UI con un campo SAPI inadecuado (no orientado a conexión). La trama UI se transmite en una conexión MAC abierta.	---	---
C_Plane/ClassA/CA/	TC_A_CA_000	ClassA_establish	Comprobar la retransmisión de la trama I de petición de establecimiento del enlace N250 veces.	---	X

Grupo de Pruebas	Caso de Prueba	Expresión de Selección	Descripción	GAP	
				FT	PT
C_Plane/ClassA/CA/	TC_A_CA_001	ClassA_establish	Verificar que la IUT, al recibir una trama de respuesta RR válida a la petición de establecimiento del enlace enviada, pasa al estado establecido.	---	X
C_Plane/ClassA/CA/	TC_A_CA_002	ClassA_re_establish_invoke	Comprobar la retransmisión de la petición de re-establecimiento del enlace N250 veces.	---	---
C_Plane/ClassA/CA/	TC_A_CA_003	ClassA_re_establish_invoke	Verificar que la IUT, al recibir una trama de respuesta RR válida a la petición de re-establecimiento del enlace enviada, pasa al estado establecido.	---	---
C_Plane/ClassA/CA/	TC_A_CA_005	ClassA_info_transfer	Verificar que la IUT confirma correctamente una trama I válida recibida dentro del temporizador <DL-04>.	X	X
C_Plane/ClassA/CA/	TC_A_CA_006	ClassA_info_transfer	Comprobar que la IUT retransmite una trama I N250 veces.	X	X
C_Plane/ClassA/CA/	TC_A_CA_007	ClassA_accept_est_req	Verificar que la IUT rechaza una petición de establecimiento de un enlace de Clase B con el envío de una trama de respuesta RR con el campo LLN puesto a “Operación de Clase A” y el bit NLF puesto a ‘1’, y pasa al estado establecido de Clase A.	X	---
C_Plane/ClassA/CA/	TC_A_CA_008	ClassA_accept_est_req	Verificar que la IUT, al recibir una petición de establecimiento, responde y pasa al estado establecido de Clase A.	X	---
C_Plane/ClassA/BV/	TC_A_BV_000	ClassA_send_and_accept_est_req	Verificar que la IUT reacciona correctamente si se produce una colisión de peticiones de establecimiento.	---	---
C_Plane/ClassA/BV/	TC_A_BV_002	ClassA_info_transfer	Verificar que la IUT acepta una trama de respuesta RR con un valor N(R) correcto como confirmación.	X	X
C_Plane/ClassA/BV/	TC_A_BV_003	ClassA_info_transfer	Verificar que la IUT acepta una trama I de comando con valores N(S) y N(R) correctos como confirmación.	X	X
C_Plane/ClassA/BV/	TC_A_BV_005	ClassA_info_transfer	Verificar que, dentro del temporizador de la fase de recuperación, la IUT acepta una trama de respuesta RR con un valor N(R) correcto como confirmación.	X	X
C_Plane/ClassA/BV/	TC_A_BV_006	ClassA_info_transfer	Verificar que, dentro del temporizador de la fase de recuperación, la IUT acepta una trama I de comando con valores N(S) y N(R) correctos como confirmación.	X	X
C_Plane/ClassA/BV/	TC_A_BV_007	Intracell_connection_ho	Verificar que la IUT gestiona correctamente el procedimiento intracelda PT para el traspaso de conexión.	*[1]	*[1]
C_Plane/ClassA/BV/	TC_A_BV_008	Intercell_connection_ho	Verificar que la IUT gestiona correctamente el procedimiento intercelda PT para el traspaso de conexión.	*[1]	*[1]
C_Plane/ClassA/BI/	TC_A_BI_000	ClassA_establish	Verificar que la IUT, en el estado de establecimiento pendiente, descarta una trama de respuesta RR clase B recibida con el bit NLF puesto a ‘1’, y retransmite la petición de establecimiento.	---	X

Grupo de Pruebas	Caso de Prueba	Expresión de Selección	Descripción	GAP	
				FT	PT
C_Plane/ClassA/BI/	TC_A_BI_001	ClassA_establish	Verificar que la IUT, en el estado de establecimiento pendiente, descarta una trama de respuesta RR recibida con el bit NLF puesto a '1' y campo N(R) inválido, y retransmite la petición de establecimiento.	---	X
C_Plane/ClassA/BI/	TC_A_BI_002	ClassA_re_establish_invoke	Verificar que la IUT, en el estado de re-establecimiento pendiente, descarta una trama de respuesta RR clase B recibida con el bit NLF puesto a '1', y retransmite la petición de re-establecimiento.	---	X
C_Plane/ClassA/BI/	TC_A_BI_003	ClassA_re_establish_invoke	Verificar que la IUT, en el estado de re-establecimiento pendiente, descarta una trama de respuesta RR recibida con el bit NLF puesto a '1' y campo N(R) inválido, y retransmite la petición de re-establecimiento.	---	X
C_Plane/ClassA/BI/	TC_A_BI_004	ClassA_info_transfer	Verificar que la IUT, en la fase de transferencia de información, descarta una trama de respuesta RR clase B recibida con el bit NLF puesto a '0' y retransmite la trama I no confirmada.	X	X
C_Plane/ClassA/BI/	TC_A_BI_005	ClassA_info_transfer	Verificar que la IUT, en la fase de transferencia de información, descarta una trama de respuesta RR recibida con el bit NLF puesto a '0' y retransmite la trama I no confirmada.	X	X
C_Plane/ClassA/BI/	TC_A_BI_006	ClassA_info_transfer	Verificar que la IUT acepta una trama I recibida con el campo N(R) inválido y, al expirar <DL-04>, retransmite la trama I no confirmada con el campo N(R) actualizado.	X	X
C_Plane/ClassA/BI/	TC_A_BI_007	ClassA_info_transfer	Al recibir una trama I con campo N(S) inválido, la IUT indica el valor N(S) esperado enviando una trama de respuesta RR o una trama I y detiene, si hace falta, el temporizador <DL-04> de acuerdo con el valor recibido del campo N(R).	X	X
C_Plane/ClassA/BI/	TC_A_BI_008	ClassA_info_transfer	Al recibir una trama I con campo N(S) inválido y campo N(R) inválido, la IUT indica el valor N(S) esperado enviando una trama de respuesta RR con el valor N(S) esperado y retransmite la trama I no confirmada.	X	X
C_Plane/ClassA/BI/	TC_A_BI_009	ClassA_info_transfer	Verificar que la IUT, en la fase de transferencia de información, descarta una trama de respuesta RR clase B recibida con el bit NLF puesto a '0' y retransmite la trama I no confirmada.	X	X
C_Plane/ClassA/BI/	TC_A_BI_011	ClassA_info_transfer	Verificar que la IUT, en la fase de recuperación temporizada, acepta una trama I recibida con el campo N(R) inválido y, al expirar <DL-04>, retransmite la trama I no confirmada con el campo N(R) actualizado.	X	X
C_Plane/ClassA/BI/	TC_A_BI_012	ClassA_info_transfer	La IUT, en la fase de recuperación temporizada y al recibir una trama I con campo N(S) inválido, indica el valor N(S) esperado enviando una trama de respuesta RR y abandona la fase de recuperación temporizada.	X	X

Grupo de Pruebas	Caso de Prueba	Expresión de Selección	Descripción	GAP	
				FT	PT
C_Plane/ClassA/BI/	TC_A_BI_013	ClassA_info_transfer	En la fase de recuperación temporizada y al recibir una trama I con campo N(S) inválido y campo N(R) inválido, la IUT indica el valor N(S) esperado enviando una trama de respuesta RR y retransmite la trama I no confirmada.	X	X
C_Plane/ClassA/BO/	TC_A_BO_000	ClassA_establish	Verificar que la IUT, en el estado de establecimiento pendiente, descarta una trama I recibida con el bit NLF puesto a '0', y retransmite la petición de establecimiento.	---	X
C_Plane/ClassA/BO/	TC_A_BO_001	ClassA_establish	Verificar que la IUT, en el estado de establecimiento pendiente, descarta una trama de respuesta RR recibida con el bit NLF puesto a '0', y retransmite la petición de establecimiento.	---	X
C_Plane/ClassA/BO/	TC_A_BO_002	ClassA_re_establish_invoke	Verificar que la IUT, en el estado de re-establecimiento pendiente, descarta una trama I recibida con el bit NLF puesto a '0', y retransmite la petición de establecimiento.	---	X
C_Plane/ClassA/BO/	TC_A_BO_003	ClassA_re_establish_invoke	Verificar que la IUT, en el estado de re-establecimiento pendiente, descarta una trama de respuesta RR recibida con el bit NLF puesto a '0', y retransmite la petición de establecimiento.	---	X
C_Plane/Lb/CA/	TC_L_CA_000	Lb_short_frame	Verificar que la IUT puede generar/recibir una trama corta de difusión (3 octetos).	X	X
C_Plane/Lb/CA/	TC_L_CA_001	Lb_long_frame	Verificar que la IUT puede generar/recibir una trama larga de difusión (3 octetos).	---	---
U_Plane/Class0/CA/	TC_0_CA_000	Class0_snd	Verificar que la IUT puede transmitir una trama correcta Clase 0 del Plano U.	X	X
U_Plane/Class0/CA/	TC_0_CA_001	Class0_rec	Verificar que la IUT puede recibir una trama correcta Clase 0 del Plano U.	X	X
U_Plane/Class1/CA/	TC_1_CA_000	Class1_snd	Verificar que la IUT puede transmitir una trama correcta Clase 1 del Plano U.	---	---
U_Plane/Class1/CA/	TC_1_CA_001	Class1_snd	Verificar que la IUT trata una trama recibida que incluye un campo RN con el bit A/N puesto a '1' como una confirmación para todas las tramas hasta e incluyendo el número de trama RN.	---	---
U_Plane/Class1/CA/	TC_1_CA_002	Class1_mandatory	Verificar que la IUT confirma correctamente trama(s) recibida(s) con lo(s) número(s) de secuencia apropiado(s). (Trama(s) en secuencia).	---	---
U_Plane/Class1/BV/	TC_1_BV_000	Class1_snd	Verificar que la IUT desconecta el enlace del Plano U al expirar el temporizador <DLU-01> sin recibir la confirmación solicitada.	---	---
U_Plane/Class1/BV/	TC_1_BV_001	Class1_snd	Verificar que la IUT resetea el temporizador <DLU-01> al recibir una confirmación válida.	---	---
U_Plane/Class1/BV/	TC_1_BV_002	Class1_snd	Verificar que la IUT mantiene el temporizador <DLU-01> cuando el tamaño de ventana se alcanza (por tanto, deteniendo transmisiones adicionales).	---	---
U_Plane/Class1/BI/	TC_1_BI_000	Class1_mandatory	Verificar que la IUT descarta una trama recibida con el bit I/R puesto a '0'.	---	---

Grupo de Pruebas	Caso de Prueba	Expresión de Selección	Descripción	GAP	
				FT	PT
U_Plane/Class1/BI/	TC_1_BI_001	Class1_mandatory	Verificar que la IUT descarta una trama recibida con el bit A/N puesto a '0'.	---	---
U_Plane/Class1/BI/	TC_1_BI_002	Class1_mandatory	Verificar que la IUT confirma correctamente la(s) trama(s) recibida(s) con número(s) de secuencia de envío erróneo(s) después de esperar por L(R) tramas TDMA (tramas fuera de secuencia).	---	---

H.2 Lista de Pruebas para el Nivel NWK-PT

La Tabla H.4 muestra la lista de Grupos de Pruebas [EN 300 497-6] para el nivel NWK de la Terminación Portátil (PT) y la Tabla H.5 enumera los Casos de Prueba [EN 300 497-7]. La columna GAP indica con 'X' que el Grupo o Caso de Prueba correspondiente es aplicable al Perfil GAP ([ETS 300 494-2], [ETS 300 494-3]). Un '*' indica que, aunque siendo aplicable al perfil GAP, no se ha podido implementar por limitaciones del Módulo de Capa Física; el número a su lado indica alguna de las siguientes limitaciones:

(1) No poder provocar un traspaso en las placas.

(2) No disponer de los algoritmos de cifrado.

(3) No disponer de primitivas de arranque (*start*) y parada (*stop*) de cifrado en MAC.

Tabla H.4: Lista de Grupos de Pruebas del Nivel NWK-PT de DECT.

Grupo de Pruebas	Expresión de Selección	Objetivo	GAP
PT/	SENG_pt_testing	Comprobar el comportamiento del nivel NWK del PT (IUT).	X
PT/CC/	SENG_cc_support	Comprobar el comportamiento de la IUT de la máquina de estados de CC.	X
PT/CC/BV/	SENG_cc_support	Comprobar la entidad CC de la IUT en respuesta a un comportamiento sintáctica y contextualmente correcto del Sistema de Pruebas.	X
PT/CC/BV/OC/	SENG_outgoing_call	Comprobar el comportamiento de la IUT para establecer una llamada saliente.	X
PT/CC/BV/IC/	SENG_incoming_call	Comprobar el comportamiento de la IUT para establecer una llamada entrante.	X
PT/CC/BV/CI/	SENG_cc_support	Comprobar el comportamiento de la IUT en los procedimientos de transferencia de información.	X
PT/CC/BV/CR/	SENG_cc_support	Comprobar el comportamiento de la IUT para liberar una llamada saliente/entrante.	X
PT/CC/BV/RS/	SENG_crss_support	Comprobar el comportamiento de la IUT durante los procedimientos de llamada relacionados con servicios suplementarios.	X
PT/CC/BO/	SENG_cc_support	Comprobar el comportamiento de la entidad CC de la IUT en respuesta a mensajes sintácticamente correctos pero no permitidos en algunos estados de los procedimientos de CC.	X
PT/CC/BI/	SENG_cc_support	Comprobar el comportamiento de la entidad CC de la IUT en respuesta a mensajes inválidos.	X
PT/CC/TI/	SENG_cc_support	Verificar que los temporizadores de CC de la IUT tienen valores correctos y que la IUT reacciona adecuadamente al vencimiento de un temporizador.	X
PT/MM/	SENG_mm_support	Comprobar el comportamiento de la entidad MM de la IUT.	X
PT/MM/BV/	SENG_mm_support	Comprobar la entidad MM de la IUT en respuesta a un comportamiento sintáctica y contextualmente correcto del Sistema de Prueba.	X
PT/MM/BV/ID/	SENG_identity_procs	Comprobar el comportamiento de la IUT referente a los procedimientos de identidad.	X
PT/MM/BV/AU/	SENG_auth_procs	Comprobar el comportamiento de la IUT referente a los procedimientos de autentificación.	X
PT/MM/BV/LO/	SENG_location_procs	Comprobar el comportamiento de la IUT referente a los procedimientos de localización.	X
PT/MM/BV/AR/	SENG_access_rights_procs	Comprobar el comportamiento de la IUT referente a los procedimientos de derechos de acceso.	X
PT/MM/BV/KA/	SENG_key_allocat_proc	Comprobar el comportamiento de la IUT referente al procedimiento de asignación de clave.	*[2]
PT/MM/BV/CH/	SENG_ciphering_procs	Comprobar el comportamiento de la IUT referente a los procedimientos relacionados con el cifrado.	X
PT/MM/BO/	SENG_mm_support	Comprobar el comportamiento de la IUT en respuesta a mensajes sintácticamente correctos pero no permitidos en alguna fase de los procedimientos de MM.	X
PT/MM/BI/	SENG_mm_support	Comprobar el comportamiento de la IUT en respuesta a mensajes MM inválidos.	X
PT/MM/TI/	SENG_mm_support	Verificar que los temporizadores de MM de la IUT tienen valores correctos y que la IUT reacciona adecuadamente al vencimiento de un temporizador.	X
PT/ME/	SENG_llme_support	Comprobar el comportamiento de la entidad LLME de la IUT.	X

Grupo de Pruebas	Expresión de Selección	Objetivo	GAP
PT/ME/BV/	SENG_llme_support	Comprobar la entidad LLME de la IUT en respuesta a un comportamiento sintáctica y contextualmente correcto del Sistema de Prueba.	X
PT/ME/BO/	SENG_llme_support	Comprobar el comportamiento de la IUT en respuesta a mensajes sintácticamente correctos pero no permitidos en alguna fase de los procedimientos gestionados de LLME.	*[2]
PT/LC/	SENG_lce_support	Comprobar el comportamiento de la entidad LCE de la IUT.	X
PT/LC/BV/	SENG_lce_support	Comprobar la entidad LCE de la IUT en respuesta a un comportamiento sintáctica y contextualmente correcto del Sistema de Prueba.	X
PT/LC/BV/LE/	SENG_lce_co	Comprobar el comportamiento de la IUT referente a los procedimientos de establecimiento del enlace orientados a conexión.	X
PT/LC/BV/LR/	SENG_lce_co	Comprobar el comportamiento de la IUT referente a los procedimientos de liberación del enlace orientados a conexión.	X
PT/LC/BI/	SENG_lce_support	Comprobar el comportamiento de la IUT en respuesta a mensajes LCE inválidos.	X
PT/LC/TI/	SENG_lce_support	Verificar que los temporizadores de LCE de la IUT tienen valores correctos y que la IUT reacciona adecuadamente al vencimiento de un temporizador.	X
PT/IS/	SENG_ciss_support	Comprobar el comportamiento de la IUT durante los procedimientos de servicios suplementarios independientes de la llamada.	---
PT/IS/BV/	SENG_ciss_support	Comprobar la entidad CISS de la IUT en respuesta a un comportamiento sintáctica y contextualmente correcto del Sistema de Prueba.	---

Tabla H.5: Lista de Casos de Prueba del Nivel NWK-PT de DECT.

Grupo de Pruebas	Caso de Prueba	Expresión de Selección	Descripción	GAP
PT/CC/BV/OC/	TC_PT_CC_BV_OC_01	SENC_piecewise	Llamada saliente; estados T-00, T-01, T-02, T-03, T-04, T-10; marcación por pasos en T-02.	X
PT/CC/BV/OC/	TC_PT_CC_BV_OC_02	SENC_piecewise	Llamada saliente; estados T-00, T-01, T-10; marcación por pasos en T-10.	X
PT/CC/BV/OC/	TC_PT_CC_BV_OC_03	SENC_piecewise_multi_digit	Llamada saliente; estados T-00, T-01, T-02, T-10; marcación por pasos en T-02 y T-10.	X
PT/CC/BV/OC/	TC_PT_CC_BV_OC_04	SENC_normal_out_call	Llamada saliente; conexión del Plano U tras <<Progress ind>> en {CC-SETUP-ACK}.	X
PT/CC/BV/OC/	TC_PT_CC_BV_OC_06	SENC_cap_n2_support	El LT simula una FT con la que la IUT no está suscrita. El LT transmitirá las bit a40 de capacidades extendidas de capa superior puesto a '1' antes de que la ITU se enganche. La IUT puede estar o no enganchada. Verificar que la IUT puede, antes de la suscripción, realizar una transición del estado CC al estado T-00 o al estado T-10 para el establecimiento de una llamada saliente de emergencia.	---
PT/CC/BV/OC/	TC_PT_CC_BV_OC_07	SENC_cap_n2_support	El LT transmitirá las bit a40 de capacidades extendidas de capa superior puesto a '1' antes de que la ITU se enganche. Verificar que la IUT puede, cuando tiene una suscripción registrada en la FT, realizar una transición al estado CC desde los estados T-00 a T-10 para el establecimiento de una llamada saliente de emergencia.	---
PT/CC/BV/OC/	TC_PT_CC_BV_OC_50	SENC_cap_n2_support	El LT simula una FT con la que la IUT no está suscrita. El LT transmitirá las bit a40 de capacidades extendidas de capa superior puesto a '0' antes de que la ITU se enganche. Verificar que la IUT no inicia, antes de suscribirse, un establecimiento de una llamada de emergencia.	---
PT/CC/BV/IC/	TC_PT_CC_BV_IC_01	SENC_signal_cc_info	Llamada entrante; estados T-01, T-06, T-07, T-08, T-10; <<SIGNAL>> en T-07.	X
PT/CC/BV/IC/	TC_PT_CC_BV_IC_02	SENC_signal_cc_setup	Llamada entrante; estados T-01, T-06, T-07, T-08, T-10; <<SIGNAL>> en {CC-SETUP}.	X
PT/CC/BV/IC/	TC_PT_CC_BV_IC_03	SENC_progress_ind_cc_setup	Llamada entrante; conexión del Plano U tras <<Progress ind>> en {CC-SETUP}.	---
PT/CC/BV/IC/	TC_PT_CC_BV_IC_04	SENC_progress_ind_cc_info	Llamada entrante; conexión del Plano U tras <<Progress ind>> en {CC-INFO} en T-07.	---
PT/CC/BV/CI/	TC_PT_CC_BV_CI_01	SENC_normal_in_call	Alerta al usuario; llamada entrante, <<SIGNAL>> en {CC-SETUP}.	X
PT/CC/BV/CI/	TC_PT_CC_BV_CI_02	SENC_go_pulse	Ir a invocación de pulsos en T-02; llamada saliente.	X
PT/CC/BV/CI/	TC_PT_CC_BV_CI_03	SENC_go_pulse	Ir a invocación de pulsos en T-10; llamada saliente.	X
PT/CC/BV/CI/	TC_PT_CC_BV_CI_04	SENC_dialling_pause	Indicación de pausa de marcación en T-02; llamada saliente.	X
PT/CC/BV/CI/	TC_PT_CC_BV_CI_05	SENC_dialling_pause	Indicación de pausa de marcación en T-10; llamada saliente.	X

Grupo de Pruebas	Caso de Prueba	Expresión de Selección	Descripción	GAP
PT/CC/BV/CI/	TC_PT_CC_BV_CI_06	SENC_go_dtmf_dl	Ir a invocación de DTMF en T-02; longitud de tono definida; llamada saliente.	X
PT/CC/BV/CI/	TC_PT_CC_BV_CI_07	SENC_go_dtmf_dl	Ir a invocación de DTMF en T-10; longitud de tono definida; llamada saliente.	X
PT/CC/BV/CI/	TC_PT_CC_BV_CI_08	SENC_go_dtmf_il	Ir a invocación de DTMF en T-02; longitud de tono infinita; llamada saliente.	X
PT/CC/BV/CI/	TC_PT_CC_BV_CI_09	SENC_go_dtmf_il	Ir a invocación de DTMF en T-10; longitud de tono infinita; llamada saliente.	X
PT/CC/BV/CI/	TC_PT_CC_BV_CI_10	SENC_basic_digits	Llamada normal saliente; T-02; {CC-INFO}, enviando <<Multi keypad>>, “0-9, estrella, almohadilla”.	X
PT/CC/BV/CI/	TC_PT_CC_BV_CI_11	SENC_internal_call	Llamada interna.	X
PT/CC/BV/CI/	TC_PT_CC_BV_CI_12	SENC_standard_char	T-10; {CC-INFO}, manejo de caracteres estándar <<Multi display>>.	X
PT/CC/BV/CI/	TC_PT_CC_BV_CI_13	SENC_control_char	T-10; {CC-INFO}, control de caracteres estándar <<Multi display>>.	X
PT/CC/BV/CI/	TC_PT_CC_BV_CI_14	SENC_reg_recall	T-10; invocación de “Register recall”; {CC-INFO}, <<Multi keypad>>	X
PT/CC/BV/CR/	TC_PT_CC_BV_CR_01	SENC_normal_out_call	Llamada saliente normal; T-02; liberación normal iniciada por FT.	X
PT/CC/BV/CR/	TC_PT_CC_BV_CR_02	SENC_normal_out_call	Llamada saliente normal; T-03; liberación normal iniciada por FT.	X
PT/CC/BV/CR/	TC_PT_CC_BV_CR_03	SENC_normal_out_call	Llamada saliente normal; T-04; liberación normal iniciada por FT.	X
PT/CC/BV/CR/	TC_PT_CC_BV_CR_04	SENC_normal_in_call	Llamada entrante; T-08; liberación normal iniciada por FT.	X
PT/CC/BV/CR/	TC_PT_CC_BV_CR_05	SENC_normal_out_call	T-10; liberación normal iniciada por FT.	X
PT/CC/BV/CR/	TC_PT_CC_BV_CR_06	SENC_normal_out_call	T-10; liberación normal iniciada por IUT.	X
PT/CC/BV/CR/	TC_PT_CC_BV_CR_07	SENC_normal_out_call	T-01; liberación anormal iniciada por FT.	X
PT/CC/BV/CR/	TC_PT_CC_BV_CR_08	SENC_normal_out_call	T-02; liberación anormal iniciada por FT.	X
PT/CC/BV/CR/	TC_PT_CC_BV_CR_09	SENC_normal_out_call	T-10; liberación normal iniciada por FT.	X
PT/CC/BV/CR/	TC_PT_CC_BV_CR_10	SENC_partial_release	T-10; liberación parcial iniciada por FT.	X
PT/CC/BV/CR/	TC_PT_CC_BV_CR_11	SENC_partial_release	T-10; liberación parcial iniciada por IUT.	X
PT/CC/BV/RS/	TC_PT_CC_BV_RS_01	SENC_clip	T-00; llamada entrante; {CC-SETUP} con <<Calling party number>>; manejo de CLIP.	X
PT/CC/BV/HP/	TC_PT_CC_BV_HP_50	SENC_cap_n1_support	Verificar que la IUT puede realizar el procedimiento de traspaso externo de establecimiento de llamada. La información de traspaso externo habrá sido proporcionada por la IUT en los elementos de información <<Ext h/o indicador>> y <<network parameter>> en un mensaje CC-SETUP-ACK durante el establecimiento de la llamada saliente.	---
PT/CC/BV/HP/	TC_PT_CC_BV_HP_51	SENC_cap_n1_support	Verificar que la IUT puede realizar el procedimiento de traspaso externo de establecimiento de llamada. La información de traspaso externo habrá sido proporcionada por la IUT en los elementos de información <<Ext h/o indicador>> y <<network parameter>> en un mensaje CC-INFO durante el establecimiento de la llamada saliente.	---

Grupo de Pruebas	Caso de Prueba	Expresión de Selección	Descripción	GAP
PT/CC/BV/HP/	TC_PT_CC_BV_HP_52	SENC_cap_n1_support	Verificar que la IUT puede realizar el procedimiento de traspaso externo de establecimiento de llamada. La información de traspaso externo habrá sido proporcionada por la IUT en los elementos de información <<Ext h/o indicador>> y <<network parameter>> en un mensaje CC-CONNECT durante el establecimiento de la llamada saliente.	---
PT/CC/BV/HP/	TC_PT_CC_BV_HP_53	SENC_cap_n1_support	Verificar que la IUT puede realizar el procedimiento de traspaso externo de establecimiento de llamada. La información de traspaso externo habrá sido proporcionada por la IUT en los elementos de información <<Ext h/o indicador>> y <<network parameter>> en un mensaje CC-SETUP durante el establecimiento de la llamada saliente.	---
PT/CC/BV/HP/	TC_PT_CC_BV_HP_54	SENC_cap_n1_support	Verificar que la IUT puede realizar el procedimiento de traspaso externo de establecimiento de llamada. La información de traspaso externo habrá sido proporcionada por la IUT en los elementos de información <<Ext h/o indicador>> y <<network parameter>> en un mensaje CC-INFO durante el establecimiento de la llamada entrante.	---
PT/CC/BV/HP/	TC_PT_CC_BV_HP_55	SENC_cap_n1_support	Verificar que la IUT solicita los parámetros de traspaso externo en cualquier momento durante una llamada entrante si el elemento de información <<Ext h/o indicador>> proporcionado durante el establecimiento de la llamada tenía el valor OID puesto a '1'. Verificar que la IUT puede, a partir de entonces, completar con éxito el procedimiento de traspaso externo de establecimiento de llamada.	---
PT/CC/BV/HP/	TC_PT_CC_BV_HP_56	SENC_cap_n1_support	Verificar que la IUT aplica el procedimiento de retirada de la referencia de traspaso tan pronto como es posible después del establecimiento de una llamada entrante (sólo en CC activo) cuando el elemento de información <<network parameter>> no fue proporcionado. Verificar que la IUT puede, a partir de entonces, completar con éxito el procedimiento de traspaso externo de establecimiento de llamada.	---
PT/CC/BV/HP/	TC_PT_CC_BV_HP_57	SENC_cap_n1_support	Verificar que la IUT inicia el procedimiento de retirada de la referencia de traspaso tan pronto como es posible después del establecimiento de una llamada entrante (sólo en CC activo) cuando el elemento de información <<network parameter>> no fue proporcionado. Verificar que la IUT considera el procedimiento de retirada de la referencia de traspaso como fallido al recibir un MM_INFO_REJECT y que, después de este fallo, la IUT no intenta completar el procedimiento de traspaso externo de establecimiento de llamada.	---

Grupo de Pruebas	Caso de Prueba	Expresión de Selección	Descripción	GAP
PT/CC/BV/HP/	TC_PT_CC_BV_HP_58	SENC_cap_n1_support	Verificar que la IUT inicia el procedimiento de retirada de la referencia de traspaso tan pronto como es posible después del establecimiento de una llamada entrante (sólo en CC activo) cuando el elemento de información <<network parameter>> no fue proporcionado. Verificar que la IUT considera el procedimiento de retirada de la referencia de traspaso como fallido al expirar <MM_info.1> y que, después de este fallo, la IUT no intenta completar el procedimiento de traspaso externo de establecimiento de llamada.	---
PT/CC/BV/HP/	TC_PT_CC_BV_HP_59	SENC_cap_n1_support	Verificar que la IUT puede realizar el procedimiento de retirada de la referencia de traspaso y que establece con éxito el Plano U en la nueva conexión.	---
PT/CC/BV/HP/	TC_PT_CC_BV_HP_60	SENC_cap_n1_support	Verificar que la IUT puede completar el procedimiento de traspaso externo de establecimiento de llamada con éxito cuanto ocurre una liberación anormal del enlace en el enlace FP-1.	---
PT/CC/BV/HP/	TC_PT_CC_BV_HP_61	SENC_cap_n1_support	Verificar que la IUT, durante el procedimiento de traspaso externo de establecimiento de llamada, después de completar el establecimiento de llamada en FP-2, libera el enlace FP-1 usando un mensaje {CC-RELEASE} con <<release reason>> indicando “external handover release” si el temporizador N400 expira.	---
PT/CC/BV/HP/	TC_PT_CC_BV_HP_62	SENC_cap_n1_support	Verificar que, después de que el traspaso ha sido aceptado y el usuario libera la llamada, se usa el procedimiento de liberación de llamada como está definido en ETS 300 444 subcláusula 8.7 en relación a FP-2.	---
PT/CC/BV/HP/	TC_PT_CC_BV_HP_63	SENC_cap_n1_support	Verificar que, después de que el traspaso ha sido aceptado y la red libera la llamada, se usa el procedimiento de liberación de llamada como está definido en ETS 300 444 subcláusula 8.7 en relación a FP-2.	---
PT/CC/BV/HP/	TC_PT_CC_BV_HP_64	SENC_cap_n1_support	Verificar que, al recibir una indicación de liberación del enlace (FP-2) antes de que la petición de traspaso haya sido confirmada, el PT permanece conectado a FP-2.	---
PT/CC/BV/HP/	TC_PT_CC_BV_HP_65	SENC_cap_n1_pt_init_cipher	Verificar que, después de realizar el procedimiento de traspaso externo de establecimiento de llamada en una conexión cifrada, la IUT inicia y realiza con éxito el procedimiento de cifrado iniciado por PT en la nueva conexión.	---
PT/CC/BV/HP/	TC_PT_CC_BV_HP_66	SENC_cap_n1_ft_init_cipher	Verificar que, después de realizar el procedimiento de traspaso externo de establecimiento de llamada, la IUT realiza con éxito el procedimiento de cifrado iniciado por FT en la nueva conexión cuando se envía {CIPHER-REQUEST} antes que la primera llamada sea liberada por el LT.	---

Grupo de Pruebas	Caso de Prueba	Expresión de Selección	Descripción	GAP
PT/CC/BV/HP/	TC_PT_CC_BV_HP_67	SENC_cap_n1_pt_init_cipher	Verificar que, después de realizar el procedimiento de traspaso externo de establecimiento de llamada, e iniciar el procedimiento de cifrado iniciado por PT en la nueva conexión, la IUT libera el enlace en FP-2 si el cifrado iniciado por PT falla.	---
PT/CC/BV/HP/	TC_PT_CC_BV_HP_68	NEVER_SELECT	Verificar que, después de realizar el procedimiento de traspaso externo de establecimiento de llamada, e iniciar el procedimiento de cifrado iniciado por FT en la nueva conexión, la IUT libera el enlace en FP-2 si el cifrado iniciado por FT falla.	---
PT/CC/BV/HP/	TC_PT_CC_BV_HP_69	SENC_cap_n1_support	Verificar que, después de realizar el procedimiento de traspaso externo de establecimiento de llamada a una FP en un área de localización diferente, la IUT realiza inmediatamente el procedimiento de registro de localización en la nueva conexión.	---
PT/CC/BV/HP/	TC_PT_CC_BV_HP_70	SENC_cap_n1_support	Verificar que, después de N501 intentos consecutivos de traspaso externo sin éxito, la IUT espera al menos N500 segundos antes de iniciar un nuevo intento de traspaso externo.	---
PT/CC/BV/HP/	TC_PT_CC_BV_HP_71	SENC_cap_n1_support	Verificar que la IUT puede realizar el procedimiento de traspaso externo de establecimiento de llamada si el elemento de información <<Ext h/o indicador>> proporcionado durante el establecimiento de llamada indicó multitrama, número de multitrama, y sincronización PSCN en el campo SYNC.	---
PT/CC/BV/HP/	TC_PT_CC_BV_HP_72	SENC_cap_n1_support	Verificar que la IUT puede realizar el procedimiento de traspaso externo de establecimiento de llamada si el elemento de información <<Ext h/o indicador>> proporcionado durante el establecimiento de llamada indicó multitrama y sincronización PSCN en el campo SYNC.	---
PT/CC/BV/HP/	TC_PT_CC_BV_HP_73	SENC_cap_n1_min_sync	Para aquellas IUT que soporten la característica, verificar que la IUT puede realizar el procedimiento de traspaso externo de establecimiento de llamada si el elemento de información <<Ext h/o indicador>> proporcionado durante el establecimiento de llamada indicó sincronización multitrama en el campo SYNC.	---
PT/CC/BV/HP/	TC_PT_CC_BV_HP_74	SENC_cap_n1_no_sync	Para aquellas IUT que soporten la característica, verificar que la IUT puede realizar el procedimiento de traspaso externo de establecimiento de llamada si el elemento de información <<Ext h/o indicador>> proporcionado durante el establecimiento de llamada indicó ninguna sincronización en el campo SYNC.	---
PT/CC/BV/HP/	TC_PT_CC_BV_HP_75	NEVER_SELECT	Verificar que, después de realizar el procedimiento de traspaso externo de establecimiento de llamada con éxito, la IUT no intentará realizar otro traspaso externo de establecimiento de llamada hasta que expire el temporizador N500.	---

Grupo de Pruebas	Caso de Prueba	Expresión de Selección	Descripción	GAP
PT/CC/BO/	TC_PT_CC_BO_01	SENC_normal_out_call	T-03; mensaje {CC-CALL-PROC} inesperado; ignorar.	X
PT/CC/BO/	TC_PT_CC_BO_02	SENC_normal_out_call	T-19; recepción de {CC-RELEASE}; colisión de liberación; liberar la llamada.	X
PT/CC/BI/	TC_PT_CC_BI_01	SENC_normal_in_call	T-00; {CC-SETUP} con IE obligatorio que falta; responder con {CC-RELEASE-COM}.	X
PT/CC/BI/	TC_PT_CC_BI_02	SENC_normal_in_call	T-00; {CC-SETUP} con IE obligatorio erróneo; responder con {CC-RELEASE-COM}.	X
PT/CC/BI/	TC_PT_CC_BI_03	SENC_normal_in_call	T-00; mensaje similar a {CC-SETUP}, pero tipo de mensaje {CC-SETUP} no reconocido; ignorar.	X
PT/CC/TI/	TC_PT_CC_TI_01	SENC_normal_out_call	T-19; temporizador P-<CC.02> expira (margen de -10%); IUT envía {CC-RELEASE-COM}.	X
PT/CC/TI/	TC_PT_CC_TI_02	SENC_normal_out_call	Llamada saliente; temporizador P-<CC.03> expira (margen de -10%); IUT envía {CC-RELEASE-COM}.	X
PT/CC/TI/	TC_PT_CC_TI_03	SENC_normal_out_call	T-01; reinicia P-<CC.03> tras {CC-NOTIFY}.	X
PT/CC/TI/	TC_PT_CC_TI_04	SENC_normal_in_call	Llamada saliente; T-08; temporizador P-<CC.05> expira (margen de -10%); IUT envía {CC-RELEASE}.	X
PT/MM/BV/ID/	TC_PT_MM_BV_ID_01	SENC_identification	Petición de identidad; tipo IPUI solicitado; IPUI devuelto.	X
PT/MM/BV/ID/	TC_PT_MM_BV_ID_02	SENC_identification	Petición de identidad; tipo de identidad solicitado no disponible; ninguna identidad en la respuesta.	X
PT/MM/BV/ID/	TC_PT_MM_BV_ID_03	SENC_identification_resp_ind	Petición de identidad; tipo IPUI solicitado; dos IPUIs almacenados; dos IPUIs devueltos.	---
PT/MM/BV/ID/	TC_PT_MM_BV_ID_04	SENC_identification_resp_ind	Petición de identidad; tipo de identidad de portátil IPUI y tipo de identidad fija PARK solicitados; IPUI y PARK devueltos.	---
PT/MM/BV/ID/	TC_PT_MM_BV_ID_08	SENC_identification Identity	Petición; tipo PARK solicitado; PARK devuelto.	X
PT/MM/BV/AU/	TC_PT_MM_BV_AU_01	SENC_pt_auth	Autentificación de PT; IUT (PT) no tiene ningún valor ZAP almacenado ni información de la clase de servicio.	*[2]
PT/MM/BV/AU/	TC_PT_MM_BV_AU_02	SENC_pt_auth	Autentificación de PT; algoritmo solicitado inaceptable; rechazar.	X
PT/MM/BV/AU/	TC_PT_MM_BV_AU_03	SENC_zap	Autentificación de PT; IUT (PT) tiene un valor ZAP almacenado; IUT incluye el valor ZAP en la respuesta.	*[2]
PT/MM/BV/AU/	TC_PT_MM_BV_AU_04	SENC_zap	Autentificación de PT; manejo del incremento de ZAP.	*[2]
PT/MM/BV/AU/	TC_PT_MM_BV_AU_05	SENC_zap_ft_auth	Autentificación de PT; manejo del incremento de ZAP; autentificación de FT sin éxito; ZAP no se incrementa.	*[2]
PT/MM/BV/AU/	TC_PT_MM_BV_AU_06	SENC_pt_auth_dck_ft_cipher_ident	Autentificación de PT; manejo del almacenamiento de DCK.	*[2]

Grupo de Pruebas	Caso de Prueba	Expresión de Selección	Descripción	GAP
PT/MM/BV/AU/	TC_PT_MM_BV_AU_07	SENC_user_auth	Autentificación de usuario.	*[2]
PT/MM/BV/AU/	TC_PT_MM_BV_AU_08	SENC_ft_auth	Autentificación de FT; iniciada por IUT.	*[2]
PT/MM/BV/AU/	TC_PT_MM_BV_AU_09	SENC_service_class	Autentificación de PT; IUT (PT) ha almacenado la información de la clase de servicio; IUT incluye la información de la clase de servicio en la respuesta.	*[2]
PT/MM/BV/LO/	TC_PT_MM_BV_LO_01	SENC_location_reg	Registro de localización después de obtener derechos de acceso; a44 y a38 = 1 al engancharse; sin asignación de TPUI.	X
PT/MM/BV/LO/	TC_PT_MM_BV_LO_02	SENC_location_reg	Registro de localización después de obtener derechos de acceso; a44 y a38 = 1 al engancharse; asignación de TPUI.	X
PT/MM/BV/LO/	TC_PT_MM_BV_LO_03	SENC_location_reg	Registro de localización después de obtener derechos de acceso; a44 y a38 = 0 al engancharse; IUT no realiza el registro de localización.	X
PT/MM/BV/LO/	TC_PT_MM_BV_LO_04	SENC_location_reg	Registro de localización; sin actividades CC; el área de localización cambia; a38 = 1 al engancharse y al comienzo del procedimiento; sin asignación de TPUI.	X
PT/MM/BV/LO/	TC_PT_MM_BV_LO_05	SENC_location_reg	Sin actividades CC; apagado, encendido; petición de registro de localización.	X
PT/MM/BV/LO/	TC_PT_MM_BV_LO_06	SENC_location_reg	Registro de localización; asignación de TPUI inaceptable; rechazar.	X
PT/MM/BV/LO/	TC_PT_MM_BV_LO_07	SENC_loc_reg_identif	Registro de localización; entrando en nueva área de localización; IUT borra el TPUI antiguo – ningún TPUI devuelto en el procedimiento de identificación.	X
PT/MM/BV/LO/	TC_PT_MM_BV_LO_08	SENC_location_update	Actualización de localización sugerida por FT; registro de localización iniciado por IUT; a38 = 1 al engancharse y al inicio del procedimiento.	*[2]
PT/MM/BV/LO/	TC_PT_MM_BV_LO_09	SENC_location_update	Actualización de localización sugerida por FT; registro de localización iniciado por IUT; a38 = 1 al engancharse; a38 = 0 al inicio del procedimiento.	*[2]
PT/MM/BV/LO/	TC_PT_MM_BV_LO_10	SENC_cap_m3_support	Verificar que la IUT puede realizar correctamente el registro de localización y a continuación una llamada saliente cuando utiliza SARI como el medio para engancharse al LT.	*[2]
PT/MM/BV/LO/	TC_PT_MM_BV_LO_50	SENC_cap_n5_support	Verificar que la IUT transmite un mensaje DETACH válido hacia una FP pública al apagarse.	---
PT/MM/BV/LO/	TC_PT_MM_BV_LO_51	SENC_cap_n5_support	El LT simula 2 estaciones base (FP_1 y FP_2) con diferentes RFPIs. La IUT está suscriba a ambas y enganchada a FP_1. Verificar que la IUT, cuando está enganchada al LT (FP_1), transmite un mensaje DETACH válido a la LT (FP_1) al cambiar de suscripción a la suscripción con FP_2. El DETACH será enviado antes de iniciar el registro de localización a la FP_2.	---
PT/MM/BV/LO/	TC_PT_MM_BV_LO_52	SENC_cap_n6_support	Verificar que la IUT repite periódicamente el procedimiento de localización inmediatamente después de expirar el periodo de tiempo definido por el elemento de información <<DURATION>> en el mensaje LOCATE-ACCEPT. El LT indica “Defined time limit 1”.	---

Grupo de Pruebas	Caso de Prueba	Expresión de Selección	Descripción	GAP
PT/MM/BV/LO/	TC_PT_MM_BV_LO_53	SENC_cap_n6_support	Verificar que la IUT repite periódicamente el procedimiento de localización inmediatamente después de expirar el periodo de tiempo definido por el elemento de información <<DURATION>> en el mensaje LOCATE-ACCEPT. El LT indica “Defined time limit 2”.	---
PT/MM/BV/LO/	TC_PT_MM_BV_LO_54	SENC_cap_n6_support	Verificar que la IUT no repite periódicamente el procedimiento de localización cuando expira el periodo de tiempo definido por el elemento de información <<DURATION>> en el mensaje LOCATE-ACCEPT indica “Infinite”.	---
PT/MM/BV/LO/	TC_PT_MM_BV_LO_55	SENC_cap_n6_support	Verificar que la IUT repite periódicamente el procedimiento de localización inmediatamente después de expirar el periodo de tiempo definido por el elemento de información <<DURATION>> en el mensaje LOCATE-REJECT. El LT indica “Defined time limit 1”.	---
PT/MM/BV/LO/	TC_PT_MM_BV_LO_56	SENC_cap_n6_support	Verificar que la IUT iniciará el procedimiento de registro de localización después de engancharse al LT (FP) si la IUT pierde el enganche y no se puede enganchar otra vez dentro del periodo de tiempo definido por el elemento de información <<DURATION>> en el mensaje LOCATE-ACCEPT. El LT indica “Temporary user limit 1”.	---
PT/MM/BV/LO/	TC_PT_MM_BV_LO_57	SENC_cap_n6_support	Verificar que la IUT iniciará el procedimiento de registro de localización después de engancharse al LT (FP) si la IUT pierde el enganche y no se puede enganchar otra vez dentro del periodo de tiempo definido por el elemento de información <<DURATION>> en el mensaje LOCATE-ACCEPT. El LT indica “Temporary user limit 2”.	---
PT/MM/BV/LO/	TC_PT_MM_BV_LO_58	SENC_cap_n6_support	Verificar que la IUT no iniciará el procedimiento de registro de localización después de engancharse al LT (FP) si la IUT pierde el enganche y no se puede enganchar otra vez dentro del tiempo definido por el elemento de información <<DURATION>> en el mensaje LOCATE-ACCEPT. El LT indica “No limits”.	---
PT/MM/BV/LO/	TC_PT_MM_BV_LO_59	SENC_cap_n6_support	Verificar que la IUT borra el TPUI si la IUT abandona el estado de enganchado con ese LT (falla al recibir el PARK) durante más de T603 segundos cuando el elemento de información <<DURATION>> en el mensaje LOCATE-ACCEPT recibido durante el último registro de localización con éxito indicaba “Temporary user limit 2”.	---
PT/MM/BV/LO/	TC_PT_MM_BV_LO_60	SENC_cap_n6_support	Verificar que cuando la IUT no recibe ninguna respuesta a {LOCATE-REQUEST}, hace un nuevo intento de registro de localización después de al menos N700 y antes de N700+N800.	---

Grupo de Pruebas	Caso de Prueba	Expresión de Selección	Descripción	GAP
PT/MM/BV/LO/	TC_PT_MM_BV_LO_61	SENC_cap_m1_support	Verificar que la IUT puede realizar correctamente una localización de registro y a continuación una llamada saliente cuando utiliza TARI como el medio para engancharse al LT.	---
PT/MM/BV/AR/	TC_PT_MM_BV_AR_01	SENC_access_rights	Obtención de derechos de acceso; a44 = 1; ambos extremos usan AC.	*[2]
PT/MM/BV/AR/	TC_PT_MM_BV_AR_03	SENC_access_rights	Obtención de derechos de acceso; a44 = 0; IUT no inicia el procedimiento de obtención de derechos de acceso.	X
PT/MM/BV/AR/	TC_PT_MM_BV_AR_05	SENC_ft_terminate_ar	Rescindir los derechos de acceso; iniciado por FT; IUT (PT) puede autentificar FT.	X
PT/MM/BV/AR/	TC_PT_MM_BV_AR_06	SENC_ft_term_ar_ft_auth	Rescindir los derechos de acceso; iniciado por FT; IUT (PT) puede autentificar FT; la autentificación falla; rescisión rechazada.	*[2]
PT/MM/BV/AR/	TC_PT_MM_BV_AR_09	SENC_zap	Obtención de derechos de acceso; FT asigna el campo ZAP; IUT lo almacena.	*[2]
PT/MM/BV/AR/	TC_PT_MM_BV_AR_10	SENC_service_class	Obtención de derechos de acceso; FT asigna la clase de servicio; IUT la almacena.	*[2]
PT/MM/BV/AR/	TC_PT_MM_BV_AR_50	SENC_cap_n7_support	Verificar que la IUT realiza correctamente el procedimiento de modificación en el aire de los parámetros de usuario y que después de este procedimiento la IUT almacena los nuevos parámetros correctamente.	---
PT/MM/BV/AR/	TC_PT_MM_BV_AR_51	SENC_cap_n7_support	Verificar que la IUT no iniciará el procedimiento de obtención de derechos de acceso como respuesta a la sugerencia de modificar los derechos de acceso si el procedimiento de autentificación de la FT falla.	---
PT/MM/BV/AR/	TC_PT_MM_BV_AR_52	SENC_cap_n7_support	Verificar que la IUT no modificará los parámetros actuales de derechos de acceso si el procedimiento de obtención de derechos de acceso como respuesta a la sugerencia de modificar los derechos de acceso falla debido a un fallo del enlace.	---
PT/MM/BV/AR/	TC_PT_MM_BV_AR_53	SENC_cap_n7_support	Verificar que la IUT no modificará los parámetros actuales de derechos de acceso si el procedimiento de obtención de derechos de acceso como respuesta a la sugerencia de modificar los derechos de acceso falla debido a no haber respuesta desde el LT (FT).	---
PT/MM/BV/KA/	TC_PT_MM_BV_KA_01	SENC_key_allocate	Asignación de clave.	*[2]
PT/MM/BV/KA/	TC_PT_MM_BV_KA_02	SENC_key_allocate	Asignación de clave; <<Auth type>> inaceptable; rechazar.	*[2]
PT/MM/BV/KA/	TC_PT_MM_BV_KA_03	SENC_key_alloc_ft_auth	Asignación de clave; autentificación implícita de FT falla; la clave no se asigna.	*[2]
PT/MM/BV/CH/	TC_PT_MM_BV_CH_01	SENC_pt_cipher_on	Conmutación de cifrado; iniciada por IUT (PT); de “cipher-off” a “cipher-on”.	X
PT/MM/BV/CH/	TC_PT_MM_BV_CH_02	SENC_pt_cipher_off	Conmutación de cifrado; iniciada por IUT (PT); de “cipher-on” a “cipher-off”.	X
PT/MM/BV/CH/	TC_PT_MM_BV_CH_03	SENC_ft_cipher_on	Conmutación de cifrado; iniciada por FT; de “cipher-off” a “cipher-on”.	*[2]
PT/MM/BV/CH/	TC_PT_MM_BV_CH_04	SENC_ft_cipher_off	Conmutación de cifrado; iniciada por FT; de “cipher-on” a “cipher-off”.	*[2]
PT/MM/BV/CH/	TC_PT_MM_BV_CH_05	SENC_ft_cipher_on	Conmutación de cifrado; iniciada por FT; de “cipher-off” a “cipher-on”; clave o algoritmo no aceptables; rechazar.	X

Grupo de Pruebas	Caso de Prueba	Expresión de Selección	Descripción	GAP
PT/MM/BV/CH/	TC_PT_MM_BV_CH_08	SENC_pt_cipher_on	Conmutación de cifrado; iniciada por IUT (PT); de “cipher-off” a “cipher-on” falla; liberación del enlace.	*[3]
PT/MM/BV/CH/	TC_PT_MM_BV_CH_09	SENC_pt_cipher_on	Conmutación de cifrado; iniciada por IUT (PT); de “cipher-off” a “cipher-on”; traspaso de portadora inter-celda con éxito.	*[1,2]
PT/MM/BV/CH/	TC_PT_MM_BV_CH_10	SENC_pt_cipher_on	Conmutación de cifrado; iniciada por IUT (PT); de “cipher-off” a “cipher-on”; traspaso de portadora intra-celda con éxito.	*[1]
PT/MM/BV/CH/	TC_PT_MM_BV_CH_11	SENC_pt_cipher_off	Conmutación de cifrado; iniciada por IUT (PT); de “cipher-off” a “cipher-on”; “cipher-on” a “cipher-off” falla; liberación del enlace.	*[3]
PT/MM/BV/CH/	TC_PT_MM_BV_CH_12	SENC_ft_cipher_on	Conmutación de cifrado; iniciada por FT; de “cipher-off” a “cipher-on”; liberación del enlace.	*[2,3]
PT/MM/BV/CH/	TC_PT_MM_BV_CH_13	SENC_ft_cipher_on	Conmutación de cifrado; iniciada por FT; de “cipher-off” a “cipher-on”; traspaso de portadora inter-celda con éxito.	*[1,2]
PT/MM/BV/CH/	TC_PT_MM_BV_CH_14	SENC_ft_cipher_on	Conmutación de cifrado; iniciada por FT; de “cipher-off” a “cipher-on”; traspaso de portadora intra-celda con éxito.	*[1,2]
PT/MM/BV/CH/	TC_PT_MM_BV_CH_15	SENC_ft_cipher_off	Conmutación de cifrado; iniciada por FT; de “cipher-off” a “cipher-on”; “cipher-on” a “cipher-off” falla; liberación del enlace.	*[2,3]
PT/MM/BO/	TC_PT_MM_BO_01	SENC_access_rights_loc	Petición de registro de localización; recepción de {ACCESS-RIGHTS-ACCEPT}; inesperada, ignorar.	X
PT/MM/BI/	TC_PT_MM_BI_01	SENC_mm_general	Tipo de mensaje no reconocido; ignorar.	X
PT/MM/BI/	TC_PT_MM_BI_02	SENC_ft_cipher_on	{CIPHER-REQUEST} con <<Cipher info>> inválido; rechazar.	X
PT/MM/BI/	TC_PT_MM_BI_03	SENC_pt_auth	Autentificación de PT ; {AUTH-REQUEST} con <<RAND>> que falta; rechazar.	X
PT/MM/BI/	TC_PT_MM_BI_04	SENC_access_rights	Obtención de derechos de acceso; {ACCESS-RIGHTS-ACCEPT}, <<Portable id>> erróneo; ignorar.	X
PT/MM/TI/	TC_PT_MM_TI_01	SENC_key_alloc_ident	Asignación de clave; expira el temporizador P-<MM_auth.1> (margen de +5%).	X
PT/MM/TI/	TC_PT_MM_TI_02	NEVER_SELECT	Autentificación de FT; justo antes de que expire el temporizador P-<MM_auth.1> (margen de -10%).	---
PT/MM/TI/	TC_PT_MM_TI_03	SENC_location_reg	Registro de localización; justo antes de que expire el temporizador P-<MM_locate.1> (margen de -10%).	X
PT/MM/TI/	TC_PT_MM_TI_04	SENC_access_rights	Obtención de derechos de acceso; justo antes de que expire el temporizador P-<MM_access.1> (margen de -10%).	X
PT/MM/TI/	TC_PT_MM_TI_05	SENC_pt_cipher_on	Conmutación de cifrado; iniciado por IUT (PT); expira el temporizador P-<MM_cipher.2> (margen de -10%).	X
PT/ME/BV/	TC_PT_ME_BV_01	SENC_out_call_pt_auth	Llamada saliente; T-01; Autentificación de IUT (PT) realizada antes de contestar a la petición de establecimiento.	*[2]

Grupo de Pruebas	Caso de Prueba	Expresión de Selección	Descripción	GAP
PT/ME/BV/	TC_PT_ME_BV_02	NEVER_SELECT	Conmutación de cifrado iniciada por IUT (PT); actualización de localización; inicio del registro de localización después de “cipher off”.	---
PT/ME/BV/	TC_PT_ME_BV_03	SENC_obtain_ar_user_auth	Obtención de derechos de acceso; interrumpida por autentificación del usuario.	*[2]
PT/ME/BV/	TC_PT_ME_BV_04	SENC_obtain_ar_pt_auth	Obtención de derechos de acceso; interrumpida por autentificación de IUT (PT).	*[2]
PT/ME/BV/	TC_PT_ME_BV_05	SENC_out_call_pt_auth	Llamada saliente y autentificación de IUT (PT) en paralelo.	*[2]
PT/ME/BV/	TC_PT_ME_BV_06	SENC_out_call_ft_cipher	Llamada saliente y conmutación de cifrado iniciada por FT en paralelo.	*[2]
PT/ME/BV/	TC_PT_ME_BV_07	SENC_out_call_ft_cipher	Llamada saliente; T-01; se realiza la conmutación de cifrado iniciada por FT antes de contestar a la petición de establecimiento.	*[2]
PT/ME/BV/	TC_PT_ME_BV_09	SENC_store_dsc_ft_cipher	Cifrado activo; almacenar DCK; el Nuevo DCK no se emplea en el cifrado actual.	*[2]
PT/ME/BV/	TC_PT_ME_BV_10	SENC_out_call_loc_reg	T-10; a38 = 1; cambios de área de localización; petición de registro de localización durante la llamada o en T-00.	X
PT/ME/BV/	TC_PT_ME_BV_11	SENC_out_call_ft_term_ar	Llamada saliente; T-01; se realiza la rescisión de derechos de acceso iniciada por FT antes de contestar a la petición de establecimiento.	X
PT/ME/BV/	TC_PT_ME_BV_12	SENC_normal_out_call_onhook_aut	T-10; falla el enlace; la IUT libera la llamada.	X
PT/ME/BV/	TC_PT_ME_BV_13	SENC_obtain_ar_key_alloc	Obtención de derechos de acceso interrumpida por asignación de clave.	X
PT/ME/BO/	TC_PT_ME_BO_01	SENC_ft_auth_pt_auth	Autentificación de FT interrumpida por {AUTH-REQUEST} desde FT; ignorar.	*[2]
PT/LC/BV/LE/	TC_PT_LC_BV_LE_01	SENC_link_co_pt_cc	Establecimiento de enlace directo; iniciado por IUT.	X
PT/LC/BV/LE/	TC_PT_LC_BV_LE_02	SENC_link_co_ft_indir	Establecimiento de enlace indirecto iniciado por FT.	X
PT/LC/BV/LR/	TC_PT_LC_BV_LR_01	SENC_link_rel_maintain_mm	El enlace existe; la entidad MM deja de usar el enlace; ninguna otra entidad usa el enlace; la IUT mantiene el enlace un tiempo <LCE.02>.	X
PT/LC/BV/LR/	TC_PT_LC_BV_LR_02	SENC_link_co_pt_cc	El enlace existe; la entidad CC deja de usar el enlace; ninguna otra entidad usa el enlace; liberación normal.	X
PT/LC/BV/LR/	TC_PT_LC_BV_LR_03	SENC_link_rel_maintain_cc	El enlace existe; la entidad MM deja de usar el enlace; liberación parcial acordada; ninguna otra entidad usa el enlace; la IUT mantiene el enlace un tiempo <LCE.02>.	X
PT/LC/BI/	TC_PT_LC_BI_01	SENC_pd_ti	Error en el valor del discriminador del protocolo – servicio no soportado ; IUT lo ignora.	X
PT/LC/BI/	TC_PT_LC_BI_03	SENC_identification	{IDENTITY-REQUEST} con un identificador ilegal de transacción; ignorar.	X
PT/LC/TI/	TC_PT_LC_TI_02	SENC_link_rel_maintain_mm	MM deja de usar el enlace; ninguna otra entidad usa el enlace; el temporizador <LCE.02> expira (márgenes TSPX_lce_02 – 1000 a 10500 ms).	X

Grupo de Pruebas	Caso de Prueba	Expresión de Selección	Descripción	GAP
PT/IS/BV/	TC_PT_IS_BV_50	SENC_cap_n4_support	Verificar que la IUT, como parte de un procedimiento de indicación de mensaje en espera, al recibir un mensaje {FACILITY} conteniendo un elemento de información <<FACILITY>> que especifica “numberOfMessages” igual a 127 y “basicService” igual a “speech (1)” proporciona una indicación adecuada al usuario de que hay un mensaje de voz en espera.	---
PT/IS/BV/	TC_PT_IS_BV_51	SENC_cap_n4_support	Verificar que la IUT, como parte de un procedimiento de indicación de mensaje en espera, al recibir un mensaje {FACILITY} conteniendo un elemento de información <<FACILITY>> que especifica “numberOfMessages” igual a 127 y “basicService” igual a “teletex (33)” proporciona una indicación adecuada al usuario de que hay un mensaje de teletexto en espera.	---
PT/IS/BV/	TC_PT_IS_BV_52	SENC_cap_n4_support	Verificar que la IUT, como parte de un procedimiento de indicación de mensaje en espera, al recibir un mensaje {FACILITY} conteniendo un elemento de información <<FACILITY>> que especifica “numberOfMessages” igual a 127 y “basicService” igual a “allServices (0)” proporciona una indicación adecuada al usuario de que hay un mensaje en espera.	---
PT/IS/BV/	TC_PT_IS_BV_53	SENC_cap_n4_support	Verificar que la IUT, partiendo de un estado donde hay una indicación al usuario de que hay un mensaje en espera, desactiva la indicación cuando la IUT se apaga y enciende.	---
PT/IS/BV/	TC_PT_IS_BV_54	SENC_cap_n4_support	Verificar que la IUT, partiendo de un estado donde hay una indicación al usuario de que hay un mensaje en espera, desactiva la indicación tras recibir mensajes FACILITY conteniendo “MWIindicate” para cada servicio básico especificando “numberOfMessages” a cero.	---
PT/IS/BV/	TC_PT_IS_BV_55	SENC_cap_n4_support	Verificar que la IUT, después de recibir la siguiente secuencia de mensajes FACILITY, mantiene la indicación de mensaje en espera para mensajes desconocidos.	---
PT/IS/BV/	TC_PT_IS_BV_56	SENC_cap_n4_and_n7_support	Verificar que la IUT borrará la información de mensaje en espera al cambiar de suscripción (el par activo IPUI/PARK).	---

SPICUM
servicio de publicaciones

Apéndice

I

APÉNDICE I: DISEÑO DE LA ESTRUCTURA DE LOS SISTEMAS DE PRUEBAS PARA DECT

Este apéndice incluye el conjunto de bloques, canales, procesos, rutas de señales, listas de canales y procedimientos definidos en la subfase Diseño de la Estructura de los Sistemas de Pruebas para DECT. La información está organizada en tablas que muestran la siguiente información:

a) Para cada bloque, los procesos (o bloques) que contiene, el número de instancias permitidas de cada uno de ellos y una breve descripción de su funcionalidad.

b) Para cada proceso, las rutas de señales a que se conecta, las listas de señales que transportan, el punto de conexión de las rutas de señales y el sentido.

c) Para cada proceso, los procedimientos definidos y una breve descripción de los mismos.

La información sobre los canales y listas de señales externos a cada bloque se encuentra descrita en el Capítulo 8, Sección 8.5.1.2 (*Definición de Interfaces del Módulo de Protocolos*) y no se ha repetido en este apéndice.

En el caso de que se trate de un canal que comunica dos bloques o de una ruta de señales que comunica dos procesos, sólo se muestra en el correspondiente bloque o proceso la vía de comunicación saliente (caso de ser una vía unidireccional no aparecería la vía si fuera de entrada), con el objetivo de evitar duplicidades.

Cuando una vía de comunicación no es vertical (en el sentido de que haya una ordenación clara entre las entidades conectadas), se ha elegido su sentido de acuerdo con la colocación de las entidades conectadas en el correspondiente diagrama SDL.

La estructura de este anexo se ha organizado igual que el Apéndice J. Primero aparecen los bloques que forman parte de algún Sistema de Pruebas para la Terminación Portátil, luego los bloques que forman parte del Sistema de Pruebas para la Terminación Fija y finalmente los paquetes que contienen procedimientos utilizados por estos bloques.

I.1 Sistemas de Pruebas para la Terminación Portátil

Las siguientes Secciones muestran la información de la estructura correspondiente a los bloques que forman parte de algún Sistema de Pruebas para la Terminación Portátil.

SPICUM servicio de publicaciones

I.1.1 Bloque MAC_CCF_FT

Las siguientes tablas muestran la información sobre procesos (Tabla I.1), rutas y listas de señales (Tabla I.2) y procedimientos (Tabla I.3) declarados en el bloque MAC_CCF_FT.

Tabla I.1: Procesos del Bloque MAC_CCF_FT.

Bloque	Procesos	Instancias	Descripción
MAC_CCF_FT	BMC	1	Inicializa el Módulo de Capa Física y crea los canales de difusión. Una vez inicializado el sistema se encarga del envío de la información de difusión. Controla posibles fallos en el funcionamiento del Módulo de Capa Física.
	MBC_CTRL	1	Actúa como servidor de conexiones, creando un proceso MBC para cada conexión. Redirecciona los mensajes del Nivel DLC a la conexión correspondiente y realiza la desconexión controlada de los enlaces en caso de reinicio.
	MBC	(0,2)	Crea y gestiona una conexión de datos en la Terminación Fija. Controla la comunicación entre el Nivel DLC y el Subnivel CSF y se comunica con el Nivel de Gestión.
	MBC_SELEC	1	Recibe las señales del Módulo de Capa Física y las encamina al proceso MBC adecuado.

Tabla I.2: Rutas y listas de señales conectadas a los procesos del Bloque MAC_CCF_FT.

Procesos	Rutas de señales	Listas de señales	Conexión	Sentido
BMC	MESAP_BMC	L..LLME..BMC	MESAP_MAC_FT	→
		L..BMC..LLME		←
	MASAP_BMC	L..MASAP..DLC..MAC	MASAP_FT	↓
	CCFLINSER_BMC	L..BMC..LINSER	CCF_LINSER_FT	↓
		L..LINSER..BMC		↑
	BMC_MBC_CTRL	L..BMC..MBC_CTRL	MBC_CTRL	→
MBC_CTRL	MCSAP_MBC_CTRL	L..MCSAP..DLC..MAC	MCSAP_FT	↓
		L..MCSAP..MAC..DLC		↑
	MBC_CTRL_MBC	L..MBC_CTRL..MBC	MBC	↓
	CCF_LINSER_MBC_CTRL	L..LINSER..MBC_CTRL	CCF_LINSER_FT	↑
	BMC_MBC_CTRL	L..MBC_CTRL..BMC	BMC	←
MBC	MBC_CTRL_MBC	L..MBC..MBC_CTRL	MBC_CTRL	↑
	MBC_CCFLINSER	L..MBC..LINSER	CCF_LINSER_FT	↓
	MCSAP_MBC	L..MCSAP..MAC..DLC	MCSAP_FT	↑
	MESAP_MBC	L..LLME..MBC	MESAP_MAC_FT	→
		L..MBC..LLME		←
	MBC_MBC_SELEC	L..MBC..MBC_SELEC	MBC_SELEC	↓
MBC_SELEC	MBC_MBC_SELEC	L..MBC_SELEC..MBC	MBC	↑
	CCFLINSER_MBC_SELEC	L..LINSER..MBC_SELEC	CCF_LINSER_FT	↑

SPICUM servicio de publicaciones

Tabla I.3: Procedimientos declarados en los procesos del Bloque MAC_CCF_FT.

Procesos	Procedimientos	Descripción
BMC	BmcIniFt	Inicializa variables de control a partir de la información recibida del Nivel de Gestión.
	EscMacInfoPage	Construye el mensaje de aviso a partir de la PDU de información recibida del Nivel DLC. Además, solicita al Nivel de Gestión la información de Nivel MAC que debe transmitirse.
MBC_CTRL	MbcIdIniFt	Registra un nuevo identificador de MBC.
	MbcDisFt	Elimina el identificador de MBC indicado, y construye la primitiva de desconexión a enviar.
MBC	CrearMacCondInd	Rellena la primitiva ASP_MAC_CON_IND y actualiza el identificador de MBC con el nuevo MCEI.
	ConstruirCoDataReq	Extrae los datos de la primitiva de Nivel DLC y los incluye en la primitiva a enviar a las funciones CSF.
	ComprobarOrigen	Compara el parámetro fmidpmid recibido en la indicación de conexión con el fmid almacenado, para ver si la petición se refiere a esa estación base.

I.1.2 Bloque DLC_FT

Las siguientes tablas muestran la información sobre bloques y canales (Tabla I.4), procesos (Tabla I.5), rutas y listas de señales (Tabla I.6) y procedimientos (Tabla I.7) declarados en el bloque DLC_FT.

Tabla I.4: Canales y listas de señales empleadas en el Bloque DLC_FT.

Bloques	Canales	Listas de señales	Conexión	Sentido
LINK_SERVICE_FT	SSAP	L..SAP0..DLC_FT	SAP0_FT	↓
		L..DLC..SAP0_FT		↑
	MC_SAP	L..MCSAP..DLC..MAC	MCSAP_FT	↓
		L..MCSAP..MAC..DLC		↑
	ME_SAP	L..DLC..LLME_FT	MESAP_DLC_FT	←
		L..LLME..DLC_FT		→
BROADCAST_FT	BSAP	L..SAPB..DLC_FT	SAPB_FT	↓
	MA_SAP	L..MASAP..DLC..MAC	MASAP_FT	↓

SPICUM servicio de publicaciones

Tabla I.5: Procesos del Bloque DLC_FT.

Bloque	Procesos	Instancias	Descripción
LINK_SERVICE_FT	LAPC_FT	(0,)	Implementa la entidad LAPC del Nivel DLC. Provee y controla un enlace de datos, segmenta los mensajes del Nivel de Red, detecta errores de protocolo y gestión el control de flujo.
	CTRL_FT	1	Se encarga de la interacción con el Nivel MAC para el control de las conexiones, incluyendo su establecimiento y liberación.
	SignalROUTER	1	Encamina las primitivas de Nivel NWK o MAC hacia la entidad Lc o LAPC apropiada.
	Lc_FT	(0,)	Entidad encargada de la mayoría de las funciones Lc indicadas en la norma: provisión de un enlace de datos, delimitación de tramas, gestión de la suma de comprobación y fragmentación de las tramas DLC.
BROADCAST_FT	Lb_FT	1	Es el proceso encargado del encaminamiento de la primitiva de difusión desde el Nivel NWK hasta el Nivel MAC.

Tabla I.6: Rutas y listas de señales conectadas a los procesos de los Bloques LINK_SERVICE_FT y BROADCAST_FT.

Procesos	Rutas de señales	Listas de señales	Conexión	Sentido
LAPC_FT	SSAP_LAPC	L..DLC..SAP0_FT	SSAP	↑
	LAPC_MESAP	L..MESAP..LAPC_FT	ME_SAP	→
		L..LAPC..MESAP_FT		←
	LAPC_Lc	L..LAPC..Lc_FT	Lc_FT	↓
CTRL_FT	CTRL_MCSAP	L..MCSAP..CTRL_FT	MC_SAP	↑
		L..CTRL..MCSAP_FT	MC_SAP	↓
	MESAP_CTRL	L..MESAP..CTRL_FT	ME_SAP	→
		L..CTRL..MESAP_FT		←
	CTRL_Lc	L..CTRL..Lc_FT	Lc_FT	→
SignalROUTER	SSAP_ROUTER	L..SAP0..DLC_FT	SSAP	↓
	LAPC_ROUTER	L..SAP0..DLC_FT	LAPC_FT	←
	ROUTER_Lc	L..MCSAP..Lc_FT	Lc_FT	↑
	MCSAP_ROUTER	L..MCSAP..Lc_FT	MC_SAP	↑
Lc_FT	Lc_MCSAP	L..Lc..MCSAP_FT	MC_SAP	↓
	LAPC_Lc	L..Lc..LAPC_FT	LAPC_FT	↑
Lb_FT	Lb_BSAP	L..SAPB..DLC_FT	BSAP	↓
	Lb_MASAP	L..DLC..MASAP_FT	MA_SAP	↓

Tabla I.7: Procedimientos declarados en los procesos de los Bloques LINK_SERVICE_FT y BROADCAST_FT.

Procesos	Procedimientos	Descripción
LAPC_FT	ProcesarTramaRX_FT	Chequea las tramas DLC recibidas, comprobando el tipo de trama, que los datos son correctos, etc.
	ConvDLCaNWK_FT	Invoca al procedimiento de descodificación cuando se reciben datos en una trama de información (I).
	LiberarCnx_FT	Libera los recursos asociados a un enlace.
CTRL_FT	CompruebaMACCONIND	Comprueba si la nueva conexión indicada es válida.
	ConstrLcMACCONIND	Genera la notificación de una nueva conexión MAC.
Lb_FT	ConstrMACPGREQ	Se encarga de rellenar los campos de la primitiva MAC de aviso a partir de la información de difusión del Nivel NWK.

SPICUM servicio de publicaciones

I.1.3 Bloque SUB_DLC_FT

Las siguientes tablas muestran la información sobre procesos (Tabla I.8), rutas y listas de señales (Tabla I.9) y procedimientos (Tabla I.10) declarados en el bloque SUB_DLC_FT.

Tabla I.8: Procesos del Bloque SUB_DLC_FT.

Bloque	Procesos	Instancias	Descripción
SUB_DLC_FT	ConversorTTCN	1	Realiza la conversión entre las primitivas de datos usadas por el Juego de Pruebas Ejecutables y las primitivas del Nivel MAC.
	Cuasi_Lc	1	Realiza unas funciones similares al proceso Lc_FT del bloque DLC_FT, encargándose de la delimitación y fragmentación de las tramas, verificación de la suma de comprobación y control de flujo (primitiva MAC_CO-DTR_IND).
	Signal_RTX	1	Copia las señales que no requieren procesamiento en la salida.

Tabla I.9: Rtas y listas de señales conectadas a los procesos del Bloque SUB_DLC_FT.

Procesos	Rutas de señales	Listas de señales	Conexión	Sentido
ConversorTTCN	LMAC_CONV	L..LMAC..CONV	LMAC	↓
		L..CONV..LMAC		↑
	CONV_Lc	L..CONV..LC	Cuasi_Lc	↓
Cuasi_Lc	CONV_Lc	L..LC..CONV	ConversorTTCN	↑
	MCSAP_Lc	L..LC..MCSAP_FT	MCSAP_FT	↓
		L..MCSAP..LC_FT		↑
Signal_RTX	LMAC_Sign	L..LMAC..SIGN	LMAC	↓
		L..SIGN..LMAC		↑
	MCSAP_Sign	L..SIGN..MCSAP	MCSAP_FT	↓
		L..MCSAP..SIGN		↑
	MASAP_Sign	L..SIGN..MASAP	MASAP_FT	↓

SPICUM servicio de publicaciones

Tabla I.10: Procedimientos declarados en los procesos del Bloque SUB_DLC_FT.

Procesos	Procedimientos	Descripción
ConversorTTCN	Conv_Octet_PDU_DLC	Procedimientos de conversión de las señales de datos del Nivel MAC (MAC_CO_DATA_XX) a las empleadas en los Juegos de Pruebas del Nivel DLC (MAC_DATA_XX).
	Conv_Octet_PDU_DATA_DLC	
	Conv_Octet_PDU_FILLSTRING	
	Conv_Octet_PDU_CC_SETUP_DLC	
	Conv_Octet_PDU_FU1	
	Conv_Octet_PDU_INFORMATION	
	Conv_Octet_PDU_LCE_PAGE_RESPONSE_DLC	
	Conv_Octet_PDU_LCE_SHORT_REQUEST_PAGE	
	Conv_Octet_PDU_L3_MESSAGE	
	Conv_Octet_PDU_RR	
	Conv_Primi_PDU_DLC	Procedimientos de conversión de las señales de datos de los Juegos de Pruebas del Nivel DLC (MAC_DATA_XX) a las empleadas en el Nivel MAC (MAC_CO_DATA_XX).
	CodifCabecera_SD	
	Codif_RR_SD	
	Rellenar_SD	
	ConvDLCaNWK_SD	
	CodifDatNwkHeader_SD	
	CodifPDUCCSetup_SD	
	CodifPDULCEPageR_SD	
	CodifDatFixedID_SD	
	CodifPDUL3Msg_SD	
	CodifDatBasicService_SD	
	CodifDatNwkAssId_SD	
	CodifDatCipherInfo_SD	
Cuasi_Lc	Ajustar_SD	Procedimientos para fragmentar y recombinar tramas de Nivel DLC.
	Frag_Trama_SD	
	Inic_Cola_SD	
	Comprobar_SD	

I.1.4 Bloque LLME_MAC_FT

Las siguientes tablas muestran la información sobre procesos (Tabla I.11), rutas y listas de señales (Tabla I.12) y procedimientos (Tabla I.13) declarados en el bloque LLME_MAC_FT.

Tabla I.11: Procesos del Bloque LLME_MAC_FT.

Bloque	Procesos	Instancias	Descripción
LLME_MAC_FT	LLME_MAC_FT	1	Realiza las tareas de gestión del Nivel MAC, recibiendo y almacenando la información que le envía éste, comprobando los datos y determinando si está permitido establecer una nueva conexión.

Tabla I.12: Rutas y listas de señales conectadas a los procesos del Bloque LLME_MAC_FT.

Procesos	Rutas de señales	Listas de señales	Conexión	Sentido
LLME_MAC_FT	TTCN_LLME_MAC_FT	L..TTCN..LLME_MAC_FT	TTCN_LLME_MAC_FT	↓
		L..LLME_MAC..TTCN_FT		↑
	ENV_LLME_MAC_FT	L..ENV..LLME_FT	ENV_LLME_FT	↓
	MESAP_LLME_MAC_FT	L..LLME..MAC_FT	MESAP_MAC_FT	→
		L..MAC..LLME_FT		←

SPICUM servicio de publicaciones

Tabla I.13: Procedimientos declarados en los procesos del Bloque LLME_MAC_FT.

Procesos	Procedimientos	Descripción
LLME_MAC_FT	NuevaConexion	Busca una posición libre en la lista de procesos MBC e incluye en ella los datos de la nueva conexión.
	NuevoMcei	Devuelve una identidad mcei que no está siendo utilizado para una nueva conexión.
	FinConexion	Elimina de la lista de conexiones el identificador de conexión que se le indique.
	ControlHandover	Comprueba que la Terminación Portátil que quiere realizar un traspaso ya tiene abierta una conexión.
	IniLlmeFt	Lee los valores de configuración utilizados en el Nivel MAC e inicializa las variables de control.
	CrearMacMeConAllReq	Construye el mensaje de permiso de conexión que se envía al Nivel MAC y actualiza las listas de conexiones, canales ocupados y canales libres.
	ConsultaFpCap	Lee el valor del bit indicado en la información de alto nivel que se envía como configuración a la Terminación Fija y lo remite al Subsistema de Pruebas.
	CambiaFpCap	Modifica el valor del bit indicado en la información de alto nivel que se envía como configuración a la Terminación Fija.

I.1.5 Bloque LLME_FT

Las siguientes tablas muestran la información sobre procesos (Tabla I.14), rutas y listas de señales (Tabla I.15) y procedimientos (Tabla I.16) declarados en el bloque LLME_FT.

Tabla I.14: Procesos del Bloque LLME_FT.

Bloque	Procesos	Instancias	Descripción
LLME_FT	LLME_DLC_FT	1	Se encarga de las funciones de gestión del Nivel DLC: gestión de las conexiones MAC y almacenamiento y provisión de parámetros (ej: identidades). Gestiona dos conexiones DLC y hasta dos conexiones MAC para cada enlace.
	LLME_MAC_FT	1	Realiza las tareas de gestión del Nivel MAC, recibiendo y almacenando la información que le envía éste, comprobando los datos y determinando si está permitido establecer una nueva conexión.

Tabla I.15: Rutas y listas de señales conectadas a los procesos del Bloque LLME_FT.

Procesos	Rutas de señales	Listas de señales	Conexión	Sentido
LLME_DLC_FT	TTCN_LLME_DLC_FT	L..TTCN..LLME_DLC_FT	TTCN_LLME_DLC_FT	↓
		L..LLME_DLC..TTCN_FT		↑
	MESAP_LLME_DLC_FT	L..LLME..DLC_FT	MESAP_DLC_FT	→
		L..DLC..LLME_FT		←
LLME_MAC_FT	TTCN_LLME_MAC_FT	L..TTCN..LLME_MAC_FT	TTCN_LLME_MAC_FT	↓
		L..LLME_MAC..TTCN_FT		↑
	MESAP_LLME_MAC_FT	L..LLME..MAC_FT	MESAP_MAC_FT	→
		L..MAC..LLME_FT		←
	ENV_LLME_MAC_FT	L..ENV..LLME_FT	ENV_LLME_FT	↓

SPICUM servicio de publicaciones

Tabla I.16: Procedimientos declarados en los procesos del Bloque LLME_FT.

Procesos	Procedimientos	Descripción
LLME_DLC_FT	ExtraerConID	Elimina el registro correspondiente a la conexión indicada.
	NuevoConID	Asigna en la lista de conexiones un nuevo registro para la conexión entrante, que incluye las identidades empleadas por la conexión.
	ModificarConID	Reemplaza el identificador de un registro de la lista de conexiones por el nuevo identificador.
LLME_MAC_FT	NuevaConexion	Busca una posición libre en la lista de procesos MBC e incluye en ella los datos de la nueva conexión.
	NuevoMcei	Devuelve una identidad mcei que no está siendo utilizado para una nueva conexión.
	FinConexion	Elimina de la lista de conexiones el identificador de conexión que se le indique.
	ControlHandover	Comprueba que la Terminación Portátil que quiere realizar un traspaso ya tiene abierta una conexión.
	IniLlmeFt	Lee los valores de configuración utilizados en el Nivel MAC e inicializa las variables de control.
	CrearMacMeConAllReq	Construye el mensaje de permiso de conexión que se envía al Nivel MAC y actualiza las listas de conexiones, canales ocupados y canales libres.
	ConsultaFpCap	Lee el valor del bit indicado en la información de alto nivel que se envía como configuración a la Terminación Fija y lo remite al Módulo de Pruebas.
	CambiaFpCap	Modifica el valor del bit indicado en la información de alto nivel que se envía como configuración a la Terminación Fija.

I.1.6 Bloque AJUSTE_TIPOS_FT

Las siguientes tablas muestran la información sobre procesos (Tabla I.17), rutas y listas de señales (Tabla I.18) declarados en el bloque AJUSTE_TIPOS_FT. En este bloque no se declara ningún procedimiento.

Tabla I.17: Procesos del Bloque AJUSTE_TIPOS_FT.

Bloque	Procesos	Instancias	Descripción
AJUSTE_TIPOS_FT	AJUSTE_TIPOS_FT	1	Convierte algunos tipos de datos de las Pruebas que son distintos según la Terminación para poder usar los mismos tipos internamente.

Tabla I.18: Rutas y listas de señales conectadas a los procesos del Bloque AJUSTE_TIPOS_FT.

Procesos	Rutas de señales	Listas de señales	Conexión	Sentido
AJUSTE_TIPOS_FT	DLS_AJUSTE	L..SAP0..DLC_FT	TTCN_NWK_SAP0_FT	↓
		L..DLC..SAP0_FT		↑
	DLB_AJUSTE	L..SAPB..DLC_FT	TTCN_NWK_SAPB_FT	↓
	SAP0_AJUSTE	L..SAP0..DLC_FT	SAP0_FT	↓
		L..DLC..SAP0_FT		↑
	SAPB_AJUSTE	L..SAPB..DLC_FT	SAPB_FT	↓

SPICUM servicio de publicaciones

I.1.7 Bloque LINSER_FT

Las siguientes tablas muestran la información sobre procesos (Tabla I.19), rutas y listas de señales (Tabla I.20) y procedimientos (Tabla I.21) declarados en el bloque LINSER_FT.

Tabla I.19: Procesos del Bloque LINSER_FT.

Bloque	Procesos	Instancias	Descripción
LINSER_FT	LINSER_FT	1	Implementa la comunicación entre la funciones CCF y el Módulo de Capa Física.

Tabla I.20: Rutas y listas de señales conectadas a los procesos del Bloque LINSER_FT.

Procesos	Rutas de señales	Listas de señales	Conexión	Sentido
LINSER_FT	CCF_LINSER	L..CCF..LINSER	CCF_LINSER_FT	↓
		L..LINSER..CCF		↑

Tabla I.21: Procedimientos declarados en los procesos del Bloque LINSER_FT.

Procesos	Procedimientos	Descripción
LINSER_FT	Reset_FT	Resetea el Módulo de Capa Física de la Terminación Fija.

I.2 Sistema de Pruebas para la Terminación Fija

Las siguientes Secciones muestran la información de la estructura correspondiente a los bloques que forman parte de algún Sistema de Prueba para la Terminación Fija.

I.2.1 Bloque MAC_CCF_PT

Las siguientes tablas muestran la información sobre procesos (Tabla I.22), rutas y listas de señales (Tabla I.23) y procedimientos (Tabla I.24) declarados en el bloque MAC_CCF_PT.

Tabla I.22: Procesos del Bloque MAC_CCF_PT.

Bloque	Procesos	Instancias	Descripción
MAC_CCF_PT	BMC	1	Recibe la información de difusión transmitida por la Terminación Fija.
	MBC_CTRL	1	Actúa como servidor de conexiones, creando procesos de tipo MBC. Redirecciona los mensajes del Nivel DLC a la instancia MBC correspondiente y realiza la desconexión controlada de los enlaces al reinicializar el Nivel MAC.
	MBC	(0,2)	Gestiona la transferencia de datos en una conexión.
	MBC_SELEC	1	Recibe las señales del Módulo de Capa Física y las encamina a la instancia MBC adecuada.

SPICUM servicio de publicaciones

Tabla I.23: Rutas y listas de señales conectadas a los procesos del Bloque MAC_CCF_PT.

Procesos	Rutas de señales	Listas de señales	Conexión	Sentido
BMC	MESAP_BMC	L..LLME..BMC	MESAP_MAC_PT	→
		L..BMC..LLME		←
	MASAP_BMC	L..MASAP..MAC..DLC	MASAP_PT	↑
	CCFLINSER_BMC	L..BMC..LINSER	CCF_LINSER_PT	↓
		L..LINSER..BMC		↑
	BMC_MBC_CTRL	L..BMC..MBC_CTRL	MBC_CTRL	→
MBC_CTRL	MCSAP_MBC_CTRL	L..MCSAP..DLC..MAC	MCSAP_PT	↓
		L..MCSAP..MAC..DLC		↑
	MBC_CTRL_MBC	L..MBC_CTRL..MBC	MBC	↓
	CCF_LINSER_MBC_CTRL	L..MBC_CTRL..LINSER	CCF_LINSER_PT	↓
		L..LINSER..MBC_CTRL		↑
	BMC_MBC_CTRL	L..MBC_CTRL..BMC	MBC_CTRL	←
	MESAP_MBC_CTRL	L..LLME..MBC_CTRL	MESAP_MAC_PT	→
		L..MBC_CTRL..LLME		←
MBC	MBC_CTRL_MBC	L..MBC..MBC_CTRL	MBC_CTRL	↑
	MBC_CCFLINSER	L..MBC..LINSER	CCF_LINSER_PT	↓
	MCSAP_MBC	L..MCSAP..MAC..DLC	MCSAP_PT	↑
	MBC_MBC_SELEC	L..MBC..MBC_SELEC	MBC_SELEC	↓
MBC_SELEC	MBC_MBC_SELEC	L..MBC_SELEC..MBC	MBC	↑
	CCFLINSER_MBC_SELEC	L..LINSER..MBC_SELEC	CCF_LINSER_PT	↑

Tabla I.24: Procedimientos declarados en los procesos del Bloque MAC_CCF_PT.

Procesos	Procedimientos	Descripción
BMC	LeeNt	Lee la identidad RFPI recibida en el mensaje CSF_NT_IND y extrae las identidades FMID, PARI y RPN.
	LeeFpCap	Comprueba que los campos de información del Nivel MAC y de alto nivel cumplen los requisitos del Perfil GAP.
	LeeStatInfo	Lee la información estática del sistema transmitida por la Terminación Fija y la almacena.
	ConstruirBs	Reenvía al Nivel DLC los datos recibidos en la primitiva CSF_PAGE_SHORT_IND.
	BmcIni	Inicializa variables del proceso BMC.
	SiguienteCn	Indica la siguiente portadora a escanear, dentro de los valores permitidos.
	ConfirmarConexion	Compara la información recibida en las primitivas CSF_NT_IND, CSF_STAT_IND y CSF_FP_CAP_IND durante el procedimiento de enganche antes y después de crear un Control de Portadora (DBC) en el Módulo de Capa Física. Para considerar el procedimiento completado, esta información debe ser idéntica.
MBC_CTRL	MbcIdIniPt	Registra un nuevo identificador de MBC.
	PedirMacMeConAllReq	Solicita del Nivel de Gestión la autorización para aceptar una conexión.
	TipoConexion	Devuelve el tipo de conexión según las variables de control.
	CrearDbc	Solicita al BMC que se cree un Control de Portadora (DBC). Devuelve el resultado de la operación.
	MbcDis	Elimina el identificador de MBC indicado, y construye las primitivas de desconexión a enviar.
MBC	ConstruirTbcReq	A partir de la primitiva de petición de conexión, obtiene los datos a incluir en la señal CSF_TBC_REQ para crear un canal de tráfico.
	ConstruirCoDataReq	Extrae los datos de la primitiva de Nivel DLC y los incluye en la primitiva a enviar a las funciones CSF.

SPICUM servicio de publicaciones

I.2.2 Bloque SUB_DLC_PT

Las siguientes tablas muestran la información sobre procesos (Tabla I.25), rutas y listas de señales (Tabla I.26) y procedimientos (Tabla I.27) declarados en el bloque SUB_DLC_PT.

Tabla I.25: Procesos del Bloque SUB_DLC_PT.

Bloque	Procesos	Instancias	Descripción
SUB_DLC_PT	ConversorTTCN	1	Realiza la conversión entre las primitivas de datos usadas por el Juego de Pruebas y las primitivas del Nivel MAC.
	Cuasi_Lc	1	Realiza unas funciones similares al proceso Lc_FT del bloque DLC_FT, encargándose de la delimitación y fragmentación de las tramas, verificación de la suma de comprobación y control de flujo (primitiva MAC_CO-DTR_IND).
	Signal_RTX	1	Copia las señales que no requieren procesamiento en la salida.

Tabla I.26: Rutas y listas de señales conectadas a los procesos del Bloque SUB_DLC_PT.

Procesos	Rutas de señales	Listas de señales	Conexión	Sentido
ConversorTTCN	LMAC_CONV	L..LMAC..CONV	LMAC	↓
		L..CONV..LMAC		↑
	CONV_LC	L..CONV..LC	Cuasi_Lc	↓
Cuasi_Lc	CONV_LC	L..LC..CONV	ConversorTTCN	↑
	MCSAP_Lc	L..LC..MCSAP_PT	MCSAP_PT	↓
		L..MCSAP..LC_PT		↑
Signal_RTX	LMAC_Sign	L..LMAC..SIGN	LMAC	↓
		L..SIGN..LMAC		↑
	MCSAP_Sign	L..SIGN..MCSAP	MCSAP_PT	↓
		L..MCSAP..SIGN		↑
	MASAP_Sign	L..MASAP..SIGN	MASAP_PT	↑

SPICUM servicio de publicaciones

Tabla I.27: Procedimientos declarados en los procesos del Bloque SUB_DLC_PT.

Procesos	Procedimientos	Descripción
ConversorTTCN	Conv_Octet_PDU_DLC	Procedimientos de conversión de las señales de datos del Nivel MAC (MAC_CO_DATA_XX) a las empleadas en los Juegos de Pruebas del Nivel DLC (MAC_DATA_XX).
	Conv_Octet_PDU_DATA_DLC	
	Conv_Octet_PDU_FILLSTRING	
	Conv_Octet_PDU_CC_SETUP_DLC	
	Conv_Octet_PDU_FU1	
	Conv_Octet_PDU_INFORMATION	
	Conv_Octet_PDU_LCE_PAGE_RESPONSE_DLC	
	Conv_Octet_PDU_LCE_SHORT_REQUEST_PAGE	
	Conv_Octet_PDU_L3_MESSAGE	
	Conv_Octet_PDU_RR	
	Conv_PDU_DLC_Octet	Procedimientos de conversión de las señales de datos de los Juegos de Pruebas del Nivel DLC (MAC_DATA_XX) a las empleadas en el Nivel MAC (MAC_CO_DATA_XX).
	CodifCabecera_SD	
	Codif_RR_SD	
	Rellenar_SD	
	ConvDLCaNWK_SD	
	CodifDatNwkHeader_SD	
	CodifPDUCCSetup_SD	
	CodifPDULCEPageR_SD	
	CodifDatFixedID_SD	
	CodifPDUL3Msg_SD	
	CodifDatBasicService_SD	
	CodifDatNwkAssId_SD	
	CodifDatCipherInfo_SD	
Cuasi_Lc	Ajustar_SD	Procedimientos para fragmentar y recombinar tramas de Nivel DLC.
	Frag_Trama_SD	
	Inic_Cola_SD	
	Comprobar_SD	

I.2.3 Bloque LLME_MAC_PT

Las siguientes tablas muestran la información sobre procesos (Tabla I.28), rutas y listas de señales (Tabla I.29) y procedimientos (Tabla I.30) declarados en el bloque LLME_MAC_PT.

Tabla I.28: Procesos del Bloque LLME_MAC_PT.

Bloque	Procesos	Instancias	Descripción
LLME_MAC_PT	LLME_MAC_PT	1	Realiza las tareas de gestión del Nivel MAC, recibiendo y almacenando la información que le envía éste, comprobando los datos y determinando si está permitido establecer una nueva conexión.

Tabla I.29: Rutas y listas de señales conectadas a los procesos del Bloque LLME_MAC_PT.

Procesos	Rutas de señales	Listas de señales	Conexión	Sentido
LLME_MAC_PT	TTCN_LLME_MAC_PT	L..TTCN..LLME_MAC_PT	TTCN_LLME_MAC_PT	↓
		L..LLME_MAC..TTCN_PT		↑
	ENV_LLME_MAC_PT	L..ENV..LLME_PT	ENV_LLME_PT	↓
	MESAP_LLME_MAC_PT	L..LLME..MAC_PT	MESAP_MAC_PT	→
		L..MAC..LLME_PT		←

SPICUM servicio de publicaciones

Tabla I.30: Procedimientos declarados en los procesos del Bloque `LLME_MAC_PT`.

Procesos	Procedimientos	Descripción
LLME_MAC_FT	NuevaConexionPt	Busca una posición libre en la lista de procesos MBC e incluye en ella los datos de la nueva conexión.
	ResetMbcIdList	Elimina de la lista todas las conexiones activas.
	TratarBlindSlot	Recibe la máscara de 12 bits que indica si un intervalo está disponible o no (información de intervalo ciego) que transmite la Terminación Fija.
	FinConexionPt	Elimina de la lista de conexiones el identificador de conexión que se le indique.
	GuardarInfo	Recibe y almacena los derechos de acceso (PARI), el identificador (RPN) y las capacidades (fpcap) de la Terminación Fija.
	ConsultaFpCap	Lee el valor del bit indicado en la información de alto nivel que se envía como configuración a la Terminación Fija y lo remite al Subsistema de Pruebas.
	CambiaFpCap	Modifica el valor del bit indicado en la información de alto nivel que se envía como configuración a la Terminación Fija.

I.2.4 Bloque `LINSER_PT`

Las siguientes tablas muestran la información sobre procesos (Tabla I.31), rutas y listas de señales (Tabla I.32) y procedimientos (Tabla I.33) declarados en el bloque `LINSER_PT`.

Tabla I.31: Procesos del Bloque `LINSER_PT`.

Bloque	Procesos	Instancias	Descripción
LINSER_PT	LINSER_PT	1	Implementa la comunicación entre la funciones CCF y el Módulo de Capa Física.

Tabla I.32: Rutas y listas de señales conectadas a los procesos del Bloque `LINSER_PT`.

Procesos	Rutas de señales	Listas de señales	Conexión	Sentido
LINSER_PT	CCF_LINSER	L..CCF..LINSER	CCF_LINSER_PT	↓
		L..LINSER..CCF		↑

Tabla I.33: Procedimientos declarados en los procesos del Bloque `LINSER_PT`.

Procesos	Procedimientos	Descripción
LINSER_PT	Reset_PT	Resetea el Módulo de Capa Física de la Terminación Portátil.

I.3 Paquetes

En esta sección se muestran los procedimientos utilizados por los bloques presentados en la sección anterior que se han modelado dentro de algún paquete; estos

SPICUM servicio de publicaciones

procedimientos se han organizado según el paquete al que pertenecen. El objetivo de cada uno de los paquetes utilizados es el siguiente:

- `Comun_DLC`: Incluye los procedimientos comunes a las Terminaciones PT y FT del Nivel DLC, así como los parámetros de configuración del Nivel DLC. Incluye procedimientos para la codificación y descodificación de PDUs del Nivel de Red.
- `Comun_MAC`: Este paquete incluye procedimientos comunes a las Terminaciones PT y FT utilizados para la gestión de las listas de conexiones y los parámetros de configuración del Nivel MAC.
- `Esc_Lec_Serie`: Almacena procedimientos de escritura/lectura de datos para la comunicación con Módulo de Capa Física. También hay procedimientos de conversión de datos para esta interfaz.
- `Sub_DLC`: Contiene los procesos de adaptación de primitivas entre el Subsistema de Pruebas y el Nivel MAC. Se emplea en los Módulos de Protocolos de Sistemas de Pruebas del Nivel DLC de ambas Terminaciones.

I.3.1 Paquete `Comun_DLC`

Los procedimientos modelados en el paquete `Comun_DLC` se muestran en la Tabla I.34, Tabla I.35 y Tabla I.36.

Tabla I.34: Procedimientos del Paquete `Comun_DLC`.

Procedimiento	Descripción
`Bits2Int`	Convierte una cadena `Bit_String` en un valor entero.
`Devuelve_ConID_1`	Obtiene de la lista de identificadores de conexión el registro cuyo `mcei` coincide con el que se pasa como parámetro.
`Devuelve_ConID_2`	Obtiene de la lista de identificadores de conexión el registro cuyo `pid_lc` coincide con el que se pasa como parámetro.
`Devuelve_ConID_3`	Obtiene de la lista de identificadores de conexión el registro cuyo `dlei` coincide con el que se pasa como parámetro.
`Devuelve_ConID_4`	Obtiene de la lista de identificadores de conexión el registro cuyo `pmid` coincide con el que se pasa como parámetro.
`Ajustar`	Ajusta la longitud de una trama recibida para que sea múltiplo de 5 octetos.
`Comprobar`	Chequea la suma de comprobación de una trama DLC.
`CalcChecksum`	Calcula la suma de comprobación de una trama DLC.
`InicCola`	Inicializa la cola de transmisión.
`FragTrama`	Fragmenta una trama DLC en fragmentos de cinco octetos, que es el tamaño aceptado por el Nivel MAC. Incluye los fragmentos en la cola de transmisión.
`ConvCabecera`	Convierte una cabecera DLC, estructurada en campos, en una secuencia de 3 octetos, para poder formar la trama DLC.
`SacarCola`	Elimina la primera trama de la cola de transmisión de la entidad LAPC.
`CodifCabecera`	Obtiene la información de los campos de la cabecera de una trama DLC.
`Conv_Primi_PDU_NWK`	Descodifica un mensaje de Nivel de Red. Al identificar el tipo de mensaje, invoca a la función correspondiente (ver Tabla I.36).
`ConstrTramaI`	Codifica una trama I (información) y le concatena dos octetos para incluir la suma de comprobación que se calcula en la entidad `Lc`. Rellena, si hace falta, para que la longitud total de la trama sea múltiplo de 40 bits.
`ConstrTramaRR`	Codifica una trama RR (confirmación) y le concatena dos octetos para incluir la suma de comprobación que se calcula en la entidad `Lc`.
`Conv_Octet_PDU_NWK`	Codifica un mensaje de Nivel de Red. Al identificar el tipo de mensaje, invoca a la función correspondiente (ver Tabla I.35).

SPICUM servicio de publicaciones

Tabla I.35: Procedimientos de codificación de un mensaje del Nivel de Red.

Conv_Octet_ALLOCATION_TYPE	Conv_Octet_PDU_AUTH_REPLY
Conv_Octet_ALPHANUMERIC	Conv_Octet_PDU_AUTH_REQUEST
Conv_Octet_AUTH_TYPE	Conv_Octet_PDU_BROADCAST_LONG
Conv_Octet_AUTH_TYPE_LIST	Conv_Octet_PDU_CC_ALERTING
Conv_Octet_BASIC_SERVICE_NWK	Conv_Octet_PDU_CC_CALL_PROC
Conv_Octet_CALLED_PARTY_NUMBER	Conv_Octet_PDU_CC_CONNECT
Conv_Octet_CALLED_PARTY_SUBADDRESS	Conv_Octet_PDU_CC_CONNECT_ACK
Conv_Octet_CALLING_PARTY_NUMBER	Conv_Octet_PDU_CC_INFO
Conv_Octet_CALL_ATTRIBUTES	Conv_Octet_PDU_CC_IWU_INFO
Conv_Octet_CALL_ID	Conv_Octet_PDU_CC_NOTIFY
Conv_Octet_CIPHER_INFO_NWK	Conv_Octet_PDU_CC_OUT_OF_SCOPE
Conv_Octet_CONNECTION_ATTRIBUTES	Conv_Octet_PDU_CC_RELEASE
Conv_Octet_CONNECTION_ID	Conv_Octet_PDU_CC_RELEASE_COM
Conv_Octet_DATE_TIME	Conv_Octet_PDU_CC_SETUP_ACK
Conv_Octet_DELIMITER_REQUEST	Conv_Octet_PDU_CC_SETUP_NWK
Conv_Octet_DURATION_NWK	Conv_Octet_PDU_CIPHER_REJECT
Conv_Octet_END_TO_END_COMPATIBILITY	Conv_Octet_PDU_CIPHER_REQUEST
Conv_Octet_ESCAPE_FOR_EXTENSION	Conv_Octet_PDU_CIPHER_SUGGEST
Conv_Octet_ESCAPE_TO_PROPRIETARY	Conv_Octet_PDU_CISS_ANY_PDU
Conv_Octet_EXT_HO_INDICATOR	Conv_Octet_PDU_CLMS_FIXED
Conv_Octet_FACILITY	Conv_Octet_PDU_COMS_ANY_PDU
Conv_Octet_FEATURE_ACTIVATE	Conv_Octet_PDU_DETACH
Conv_Octet_FEATURE_INDICATE	Conv_Octet_PDU_IDENTITY_REPLY
Conv_Octet_FIXED_ID	Conv_Octet_PDU_IDENTITY_REPLY_FT
Conv_Octet_IDENTITY_TYPE	Conv_Octet_PDU_IDENTITY_REPLY_PT
Conv_Octet_INFO_TYPE	Conv_Octet_PDU_IDENTITY_REQUEST
Conv_Octet_IWU_ATTRIBUTES	Conv_Octet_PDU_KEY_ALLOCATE
Conv_Octet_IWU_PACKET	Conv_Octet_PDU_LCE_PAGE_REJECT
Conv_Octet_IWU_TO_IWU	Conv_Octet_PDU_LCE_PAGE_RESPONSE_NWK
Conv_Octet_KEY	Conv_Octet_PDU_LCE_REQUEST_PAGE
Conv_Octet_LOCATION_AREA	Conv_Octet_PDU_LOCATE_ACCEPT
Conv_Octet_MMS_EXT_HEADER	Conv_Octet_PDU_LOCATE_REJECT
Conv_Octet_MMS_GENERIC_HEADER	Conv_Octet_PDU_LOCATE_REQUEST
Conv_Octet_MMS_OBJECT_HEADER	Conv_Octet_PDU_MM_INFO_SUGGEST
Conv_Octet_MODEL_IDENTIFIER	Conv_Octet_PDU_MM_OUT_OF_SCOPE
Conv_Octet_MULTI_DISPLAY	Conv_Octet_PDU_NWK
Conv_Octet_MULTI_KEYPAD	Conv_Octet_PDU_TEMPORARY_ID_ASSIGN_ACK
Conv_Octet_NETWORK_ASSIGNED_ID	Conv_Octet_PDU_TEMPORARY_ID_ASSIGN_REJECT
Conv_Octet_NETWORK_HEADER	Conv_Octet_PORTABLE_ID
Conv_Octet_NETWORK_PARAMETER	Conv_Octet_PROGRESS_INDICATOR
Conv_Octet_PDU_ACCESS_RIGHTS_ACCEPT	Conv_Octet_RAND
Conv_Octet_PDU_ACCESS_RIGHTS_REJECT	Conv_Octet_RATE_PARAMETERS
Conv_Octet_PDU_ACCESS_RIGHTS_REQUEST	Conv_Octet_REJECT_REASON
Conv_Octet_PDU_ACCESS_RIGHTS_TERM_ACCEPT	Conv_Octet_RELEASE_REASON
Conv_Octet_PDU_ACCESS_RIGHTS_TERM_REJECT	Conv_Octet_REPEAT_INDICATOR
Conv_Octet_PDU_ACCESS_RIGHTS_TERM_REQUEST	Conv_Octet_RES_NWK
Conv_Octet_PDU_AUTH_REJECT	Conv_Octet_RS
Conv_Octet_PDU_AUTH_REJECT_FT	Conv_Octet_SEGMENTED_INFO
Conv_Octet_PDU_AUTH_REJECT_PT	Conv_Octet_SENDING_COMPLETE

SPICUM servicio de publicaciones

Tabla I.36: Procedimientos de descodificación de un mensaje del Nivel de Red.

Conv_Primi_ALLOCATION_TYPE	Conv_Primi_PDU_CC_CONNECT_ACK
Conv_Primi_ALPHANUMERIC	Conv_Primi_PDU_CC_INFO
Conv_Primi_AUTH_TYPE	Conv_Primi_PDU_CC_IWU_INFO
Conv_Primi_AUTH_TYPE_LIST	Conv_Primi_PDU_CC_NOTIFY
Conv_Primi_BASIC_SERVICE_NWK	Conv_Primi_PDU_CC_OUT_OF_SCOPE
Conv_Primi_CALLED_PARTY_NUMBER	Conv_Primi_PDU_CC_RELEASE
Conv_Primi_CALLED_PARTY_SUBADDRESS	Conv_Primi_PDU_CC_RELEASE_COM
Conv_Primi_CALLING_PARTY_NUMBER	Conv_Primi_PDU_CC_SETUP_ACK
Conv_Primi_CALL_ATTRIBUTES	Conv_Primi_PDU_CC_SETUP_NWK
Conv_Primi_CALL_ID	Conv_Primi_PDU_CIPHER_REJECT
Conv_Primi_CIPHER_INFO_NWK	Conv_Primi_PDU_CIPHER_REQUEST
Conv_Primi_CONNECTION_ATTRIBUTES	Conv_Primi_PDU_CIPHER_SUGGEST
Conv_Primi_CONNECTION_ID	Conv_Primi_PDU_CISS_ANY_PDU
Conv_Primi_DATE_TIME	Conv_Primi_PDU_CLMS_FIXED
Conv_Primi_DELIMITER_REQUEST	Conv_Primi_PDU_COMS_ANY_PDU
Conv_Primi_DURATION_NWK	Conv_Primi_PDU_DETACH
Conv_Primi_END_TO_END_COMPATIBILITY	Conv_Primi_PDU_IDENTITY_REPLY
Conv_Primi_ESCAPE_FOR_EXTENSION	Conv_Primi_PDU_IDENTITY_REPLY_FT
Conv_Primi_ESCAPE_TO_PROPRIETARY	Conv_Primi_PDU_IDENTITY_REPLY_PT
Conv_Primi_EXT_HO_INDICATOR	Conv_Primi_PDU_IDENTITY_REQUEST
Conv_Primi_FACILITY	Conv_Primi_PDU_KEY_ALLOCATE
Conv_Primi_FEATURE_ACTIVATE	Conv_Primi_PDU_LCE_PAGE_REJECT
Conv_Primi_FEATURE_INDICATE	Conv_Primi_PDU_LCE_PAGE_RESPONSE_NWK
Conv_Primi_FIXED_ID	Conv_Primi_PDU_LCE_REQUEST_PAGE
Conv_Primi_IDENTITY_TYPE	Conv_Primi_PDU_LOCATE_ACCEPT
Conv_Primi_INFO_TYPE	Conv_Primi_PDU_LOCATE_REJECT
Conv_Primi_IWU_ATTRIBUTES	Conv_Primi_PDU_LOCATE_REQUEST
Conv_Primi_IWU_PACKET	Conv_Primi_PDU_MM_INFO_SUGGEST
Conv_Primi_IWU_TO_IWU	Conv_Primi_PDU_MM_OUT_OF_SCOPE
Conv_Primi_KEY	Conv_Primi_PDU_NWK
Conv_Primi_LOCATION_AREA	Conv_Primi_PDU_TEMPORARY_ID_ASSIGN_ACK
Conv_Primi_LOCATION_AREA.BAK	Conv_Primi_PDU_TEMPORARY_ID_ASSIGN_REJECT
Conv_Primi_MMS_EXT_HEADER	Conv_Primi_PORTABLE_ID
Conv_Primi_MMS_GENERIC_HEADER	Conv_Primi_PROGRESS_INDICATOR
Conv_Primi_MMS_OBJECT_HEADER	Conv_Primi_RAND
Conv_Primi_MODEL_IDENTIFIER	Conv_Primi_RATE_PARAMETERS
Conv_Primi_MULTI_DISPLAY	Conv_Primi_REJECT_REASON
Conv_Primi_MULTI_KEYPAD	Conv_Primi_RELEASE_REASON
Conv_Primi_NETWORK_ASSIGNED_ID	Conv_Primi_REPEAT_INDICATOR
Conv_Primi_NETWORK_HEADER	Conv_Primi_RES_NWK
Conv_Primi_NETWORK_PARAMETER	Conv_Primi_RS
Conv_Primi_PDU_ACCESS_RIGHTS_ACCEPT	Conv_Primi_SEGMENTED_INFO
Conv_Primi_PDU_ACCESS_RIGHTS_REJECT	Conv_Primi_SENDING_COMPLETE
Conv_Primi_PDU_ACCESS_RIGHTS_REQUEST	Conv_Primi_SERVICE_CLASS
Conv_Primi_PDU_ACCESS_RIGHTS_TERM_ACCEPT	Conv_Primi_SETUP_CAPABILITY
Conv_Primi_PDU_ACCESS_RIGHTS_TERM_REJECT	Conv_Primi_SHORT_FORMAT_ADDRESS
Conv_Primi_PDU_ACCESS_RIGHTS_TERM_REQUEST	Conv_Primi_SIGNAL_mio
Conv_Primi_PDU_AUTH_REJECT	Conv_Primi_SINGLE_DISPLAY
Conv_Primi_PDU_AUTH_REJECT_FT	Conv_Primi_SINGLE_KEYPAD
Conv_Primi_PDU_AUTH_REJECT_PT	Conv_Primi_TERMINAL_CAPABILITY
Conv_Primi_PDU_AUTH_REPLY	Conv_Primi_TERMINAL_CAPABILITY.BAK
Conv_Primi_PDU_AUTH_REQUEST	Conv_Primi_TEST_HOOK_CONTROL
Conv_Primi_PDU_BROADCAST_LONG	Conv_Primi_TIMER_RESTART
Conv_Primi_PDU_CC_ALERTING	Conv_Primi_TRANSIT_DELAY
Conv_Primi_PDU_CC_CALL_PROC	Conv_Primi_WINDOW_SIZE
Conv_Primi_PDU_CC_CONNECT	Conv_Primi_ZAP_FIELD

SPICUM servicio de publicaciones

I.3.2 Paquete SUB_DLC

El Paquete SUB_DLC incluye los tipos de proceso ConversorTTCN y Cuasi_Lc. Los procedimientos definidos en estos tipos de proceso están explicados en la Sección I.1.3.

I.3.3 Paquete Comun_MAC

Los procedimientos modelados en el paquete Comun_MAC se muestran en la Tabla I.37.

Tabla I.37: Procedimientos del Paquete Comun_MAC.

Procedimiento	Descripción
IndFromMcei	Determina el índice del elemento de la lista que tiene el mcei indicado.
IndFromSlot	Determina el índice del elemento de la lista que tiene el intervalo indicado.
IndFromPid	Determina el índice del elemento de la lista correspondiente al proceso MBC con el PId indicado.

I.3.4 Paquete Esc_Lec_Serie

Los procedimientos modelados en el paquete Esc_Lec_Serie se muestran en la Tabla I.38.

Tabla I.38: Procedimientos del Paquete Esc_Lec_Serie.

Procedimiento	Descripción
HexaInt	Devuelve el valor decimal de un número indicado como carácter ASCII.
EscEntDatPrim	Escribe un valor entero en la posición indicada de la cadena como parámetro.
EscOctDatPrim	Escribe tantos valores enteros como se indiquen a partir de la posición indicada de la cadena.
LeerEntDatPrim	Lee un número hexadecimal recibido como caracteres ASCII colocados en dos posiciones consecutivas y lo convierte en un número decimal.
LeerOctDatPrim	Lee el número en formato hexadecimal que viene a partir de la posición indicada de la cadena.

SPICUM servicio de publicaciones

SPICUM
servicio de publicaciones

Apéndice

J

APÉNDICE J: DISEÑO DETALLADO DE LOS SISTEMAS DE PRUEBAS DECT

En este Apéndice se describe el diseño detallado de los elementos incluidos en los Módulos de Protocolos de los Sistemas de Pruebas para DECT. Para organizar la descripción, se han agrupado los distintos procesos según el tipo de Terminación, Fija o Portátil, a que pertenecen y según el Nivel al que su funcionalidad está lógicamente asociada. Para cada proceso, se listan también las señales que puede recibir, las transiciones que provocan y el diagrama de estados correspondiente. Al final del Apéndice se describen los emuladores empleados en las Pruebas de Nivel.

Tabla J.1: Procesos que constituyen los bloques incluidos en los distintos Sistemas de Pruebas.

			Sistema de Pruebas		
Nivel	**Bloques**	**Procesos**	**SP_DLC_PT**	**SP_DLC_FT**	**SP_NWK_PT**
Acceso al Medio	MAC_CCF_FT	BMC	✓	-	✓
		MBC_CTRL			
		MBC			
		MBC_SELEC			
	MAC_CCF_PT	BMC	-	✓	-
		MBC_CTRL			
		MBC			
		MBC_SELEC			
	LINSER_FT	LINSER_FT	✓	-	✓
	LINSER_PT	LINSER_PT	-	✓	-
Control del Enlace	DLC_FT	CTRL_FT	-	-	✓
		LAPC_FT			
		Lc_FT			
		SignalROUTER			
		Lb_FT			
	SUB_DLC_FT	ConversorTTCN	✓	-	-
		Cuasi_Lc			
		Signal_RTX			
	SUB_DLC_PT	ConversorTTCN	-	✓	-
		Cuasi_Lc			
		Signal_RTX			
	AJUSTE_TIPOS_FT	AJUSTE_TIPOS_FT	-	-	✓
Gestión	LLME_FT	LLME_DLC_PT	-	-	✓
		LLME_MAC_PT			
	LLME_MAC_FT	LLME_MAC_FT	✓	-	-
	LLME_MAC_PT	LLME_MAC_PT	-	✓	-

SPICUM servicio de publicaciones

J.1 Sistemas de Pruebas para la Terminación Portátil

En esta Sección se describe el modelado de los procesos que integran el Módulo de Protocolos para los Sistemas de Pruebas de las Terminaciones Portátiles, `SP_DLC_PT` y `SP_NWK_PT`. La lista de estos procesos, y los bloques a que pertenecen, se muestra en la Tabla J.1. En primer lugar se describen los procesos relacionados con el Nivel de Acceso al Medio, después los relacionados con el Nivel de Control del Enlace y, finalmente, los relacionados con el Nivel de Gestión.

J.1.1 Nivel de Acceso al Medio

En este apartado se describe el comportamiento de los procesos que integran el bloque que modela el Nivel de Acceso al Medio (bloque `MAC_CCF_FT`), así como los estados que posee cada uno de ellos. Se ha incluido aquí también la descripción del proceso del bloque `LINSER_FT` por considerar que, funcionalmente, está asociado al Nivel MAC.

J.1.1.1 Proceso `BMC`

El Proceso de Control de Difusión (BMC) es el encargado de generar la información de difusión que la Terminación Fija debe transmitir periódicamente. Inicialmente el proceso se queda a la espera de que los niveles superiores le hagan llegar la información de difusión que debe transmitirse. Cuando se recibe esta información, es almacenada, se inicializa y configura el Nivel Físico y se activan los temporizadores para la transmisión periódica. En este estado se transmitirán avisos de conexión en el canal B_s cuando lo solicite el Nivel DLC; el Nivel Físico confirmará su correcta transmisión. Parte de la información que se ha de transmitir, tanto de forma periódica como en el caso de un aviso, está guardada en el Nivel de Gestión, por lo que el proceso BMC deberá comunicarse con él.

El diagrama de estados de este proceso y las transiciones posibles en cada uno de ellos para las señales válidas se muestran en la Figura J.1. El comportamiento de cada estado es el siguiente:

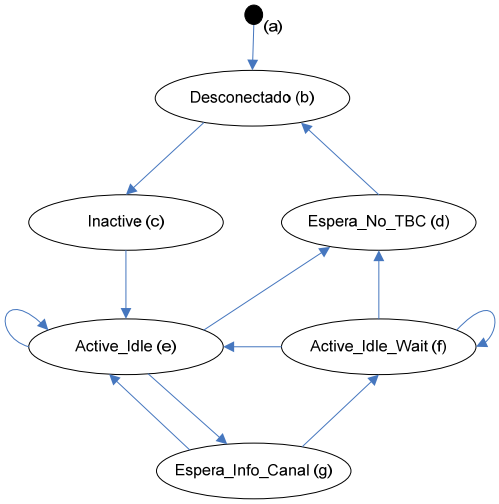
- `Desconectado`: Estado inicial en el que el proceso permanece a la espera de que el Módulo de Capa Física se inicialice adecuadamente. Cuando se recibe la señal `CSF_LINSER_START` pasa al estado `Inactive`.
- `Inactive`: Estado en el cual todavía no se ha solicitado la difusión de ninguna información. Cuando el Nivel de Gestión lo considere oportuno proporcionará (señal `MAC_ME_RFP_PRELOAD_REQ`) la información necesaria, como son la identificación de la Terminación Fija, los canales a utilizar para las conexiones y la información sobre derechos de acceso. En ese momento se activarán el Servicio de Difusión y los temporizadores que controlan la transmisión periódica de la información.
- `Espera_No_TBC`: En caso de que se produzca algún error (pérdida de sincronismo, número excesivo de errores de transmisión) en el Módulo de Capa Física se deben liberar los recursos asociados a las conexiones activas; también si hay una petición explícita desde el Nivel de Gestión. Este estado permite volver al estado `Desconectado` limpiamente, esperando la confirmación de que las conexiones han sido cerradas antes de reiniciar dicho Módulo.

SPICUM servicio de publicaciones

- `Active_Idle`: Estado en el que se está transmitiendo periódicamente la información de difusión; además, transmite los mensajes de aviso que solicite el Nivel DLC. La difusión periódica está controlada por los temporizadores `T_ZERO_PAGE`, para la información de difusión, y `T_BLIND_SLOT`, para la información de intervalos no utilizables.
- `Active_Idle_Wait`: Estado de espera de la confirmación del Módulo de Capa Física de que el mensaje de difusión solicitado ha sido enviado.
- `Espera_Info_Canal`: El proceso espera la información que necesita del Nivel de Gestión.

	Estados						
Señales	a	b	c	d	e	f	g
-	b						
CSF_LINSER_ERROR					d	d	
CSF_LINSER_START		c					
CSF_PAGE_CFM						e	
CSF_RSSI_TABLE					e	f	
CSF_TIMEOUT_IND					d	d	
MAC_ME_RESET_FT					d	d	
MAC_ME_RFP_PRELOAD_REQ			e				
MAC_ME_TX_BLIND_SLOT_REQ					g		
MAC_PAGE_REQ					g		
MAC_ME_CHAN_INFO_RES							e,f
MBC_BMC_NO_TBC				b			
T_BLIND_SLOT					g		
T_ZERO_PAGE					g	e	

(a)



(b)

Figura J.1: (a) Transiciones posibles y (b) Diagrama de estados para el proceso `BMC` de la Terminación Fija.

J.1.1.2 Proceso `MBC_CTRL`

El proceso `MBC_CTRL` es el encargado de la creación y el mantenimiento de las conexiones de tráfico. Si la petición de conexión es permitida, confirma el establecimiento y crea una instancia del proceso `MBC`, que se encargará de gestionar la nueva conexión; el proceso `MBC_CTRL` redirecciona a la instancia adecuada los mensajes del Nivel DLC. Durante una reinicialización del Subsistema Inferior, se encarga de la desconexión controlada del enlace activo.

La información de la conexión activa se almacena en una variable del tipo `MBC_ID` (Figura J.2), que contiene el PId de la instancia del proceso `MBC` asociada, el identificador de la conexión, el canal físico empleado y la identidad de la Terminación Portátil.

El diagrama de estados de este proceso y las transiciones posibles en cada uno de ellos para las señales válidas se muestran en la Figura J.3. El comportamiento de cada estado es el siguiente:

- `Active_Idle`: Es el estado inicial, en el cual no hay ninguna conexión activa y se está a la espera de una petición de conexión (`CSF_TBC_IND`) desde el Módulo

SPICUM servicio de publicaciones

de Capa Física. Cuando esta petición llega, y se autoriza, se crea una instancia del proceso MBC para gestionar la conexión.

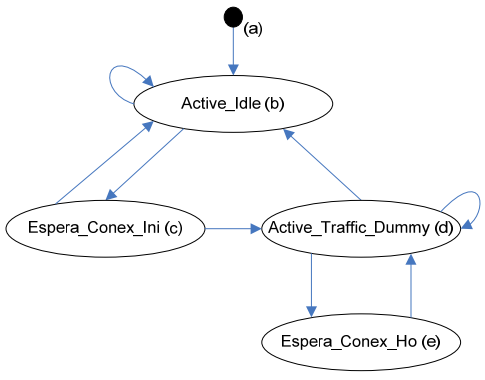
```
/* Canal físico */
newtype CANAL
  struct
    slot Integer;          /* Intervalo temporal [0..11] */
    cn   Integer;          /* Número de portadora */
endnewtype CANAL;

/* Identificador de conexión */
newtype MBC_ID
  struct
    pid Pid := null;        /* PId de la instancia MBC asociada */
    mcei MCEI := -1;        /* Identificador de la conexión */
    canal CANAL;            /* Canal físico */
    pmid Bit_String_20;     /* Identidad de la Terminación Portátil*/
endnewtype MBC_ID;
```

Figura J.2: Estructura de datos del proceso MBC_CTRL para almacenar la información de la conexión activa.

- Espera_Conex_Ini: Espera la confirmación, procedente de la instancia del proceso MBC creada, de que la conexión se ha establecido correctamente. Si se ha producido algún error, se libera la posición asignada en la lista de conexiones.
- Active_Traffic_Dummy: Se llega a este estado cuando hay una conexión activa. En este estado se encaminan las señales recibidas desde el Nivel DLC hacia la instancia MBC. Pueden recibirse peticiones de envío de datos (MAC_CO_DATA_REQ), peticiones de desconexión (MAC_DIS_REQ) y peticiones de desconexión desde la Terminación Portátil (MBC_DIS). La indicación de traspaso se recibe con la misma primitiva de la petición de conexión (CSF_TBC_IND), con el tipo indicando que se trata de un traspaso; en este caso, se crea una segunda instancia del proceso MBC y se pasa al estado Espera_Conex_Ho.
- Espera_Conex_Ho: Espera la confirmación de que el traspaso se ha realizado con éxito.

	Estados				
Señales	a	b	c	d	e
-	b				
BMC_MBC_SINC_ERR		b		d	
CSF_TBC_IND		b,c		d,e	
MAC_CO_DATA_REQ				d	
MAC_DIS_REQ				d	
MBC_DIS			b	b,d	d
MBC_MBC_CTRL_CONEX			d		d

(a)



(b)

Figura J.3: (a) Transiciones posibles y (b) Diagrama de estados para el proceso MBC_CTRL de la Terminación Fija.

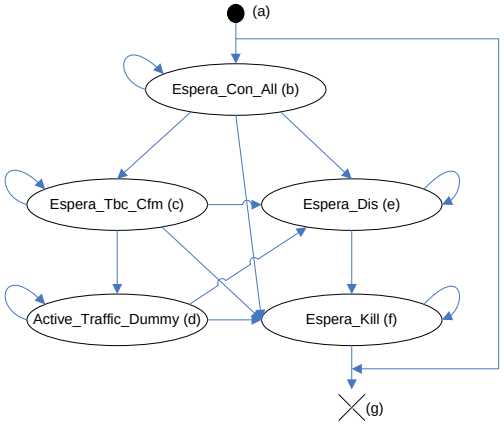
SPICUM servicio de publicaciones

J.1.1.3 Proceso MBC

El proceso MBC gestiona una conexión del Nivel MAC; las instancias de este proceso son creadas dinámicamente por el proceso MBC_CTRL cuando éste recibe una petición de conexión o de traspaso de conexión. Lo primero que hace es informar al proceso MBC_SELEC de que existe una nueva instancia (MBC_ON), con objeto de que encamine los mensajes dirigidos a ella; después, envía al Nivel de Gestión la información relativa a la conexión (MAC_ME_CON_IND). A partir de ese momento está en disposición de transmitir (MAC_CO_DATA_REQ) y recibir (CSF_CO_DATA_IND, CSF_CO_DTR_IND) datos sobre esta conexión hasta que reciba una indicación de cierre del Nivel DLC o del Nivel de Gestión.

	Estados						
Señales	a	b	c	d	e	f	g
-	b,g						
CSF_CO_DATA_IND		b	c	d	e	f	
CSF_CO_DTR_IND		b	c	d	e	f	
CSF_DIS_IND		f	f	f	f	f	
CSF_TBC_CFM		b	d	d	e	f	
MAC_CO_DATA_REQ		b	c	d	e	f	
MAC_DIS_REQ		e	e	e	e	f	
MAC_ME_CON_ALL_REQ		f,c	c	d	e	f	
MAC_ME_DIS_REQ		e	e	e	e	f	
MBC_CTRL_MBC_DIS		b	c	d	e	f	
MBC_KILL		b	c	d	e	g	
T_200		e	e	e	e	f	
T_RESP_CSF		f	f	f	f	f	

(a)



(b)

Figura J.4: (a) Transiciones posibles y (b) Diagrama de estados para el proceso MBC de la Terminación Fija.

El diagrama de estados de este proceso y las transiciones posibles en cada uno de ellos para las señales válidas se muestran en la Figura J.4. El comportamiento de cada estado es el siguiente:

- Espera_Con_All: El proceso espera la autorización para esta conexión que debe conceder el Nivel de Gestión. Si no se permite, se cierra la conexión.
- Espera_Tbc_Cfm: Se espera la confirmación de conexión desde el Módulo de Capa Física. Tras ella, se informa del identificador de conexión al proceso MBC_CTRL y se pasa al estado Active_Traffic_Dummy. Si la confirmación no se recibe en 3 segundos (temporizador T200), se pasa a liberar los recursos ya reservados.
- Active_Traffic_Dummy: En este estado existe al menos una conexión activa; mientras no se cierre, se encaminan las señales de envío y recepción de datos entre el Nivel DLC y la instancia adecuada del proceso MBC.
- Espera_Dis: Estado en el que se entra tras una petición de desconexión (CSF_TBC_REL_REQ), solicitada por el Nivel DLC, por el Nivel de Gestión o por el vencimiento de un temporizador de espera de alguna confirmación. El temporizador T_RESP_CSF asegura que, si no se recibe la indicación de desconexión del Módulo de Capa Física, se libera la conexión evitando posibles bloqueos.

SPICUM servicio de publicaciones

- `Espera_Kill`: Espera la confirmación del proceso `MBC_SELEC` de que la conexión ha sido eliminada de la lista de conexiones activas.

J.1.1.4 Proceso `MBC_SELEC`

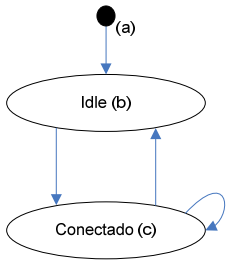
El proceso `MBC_SELEC` encamina las señales recibidas del Módulo de Capa Física a la instancia adecuada del proceso `MBC`. Este proceso encamina las señales `CSF_CO_DATA_IND`, `CSF_CO_DTR_IND`, `CSF_TBC_CFM` y `CSF_DIS_IND`.

El diagrama de estados de este proceso y las transiciones posibles en cada uno de ellos para las señales válidas se muestran en la Figura J.5. El comportamiento de cada estado es el siguiente:

- `Idle`: Es el estado inicial, a la espera de recibir la indicación de activar una nueva conexión.
- `Conectado`: En este estado, encamina las señales recibidas por la Terminación Fija a la instancia adecuada del proceso `MBC`. Cuando una instancia termina, lo indica mediante la señal `MBC_OFF`. Caso de no haber más conexiones activas, se vuelve al estado `Idle`.

	Estados		
Señales	a	b	c
-	b		
CSF_CO_DATA_IND			c
CSF_CO_DTR_IND			c
CSF_DIS_IND			c
CSF_TBC_CFM			c
MBC_OFF			b,c
MBC_ON		c	c

(a)



(b)

Figura J.5: (a) Transiciones posibles y (b) Diagrama de estados para el proceso `MBC_SELEC` de la Terminación Fija.

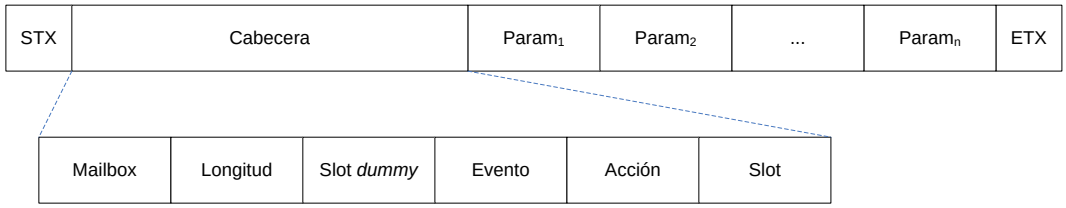
J.1.1.5 Proceso `LINSER_FT`

El proceso `LINSER_FT` es el encargado de realizar la comunicación entre el Módulo de Protocolos y el Módulo de Capa Física, llevando a cabo la codificación y descodificación de las señales intercambiadas. Este proceso hace uso del manejador descrito en el Capítulo 8, Sección 8.5.3.2.1, que accede físicamente a la interfaz serie del Módulo de Capa Física.

La primera acción es la inicialización de la interfaz de comunicación. Después, crea dos hilos, uno para recibir y otro para enviar, cada uno de los cuales dispone de una cola de señales. De esta forma, el proceso no necesita estar constantemente pendiente de la interfaz serie. Tras esta inicialización, el proceso pasa al estado `Espera_Trama` en el cual comprueba periódicamente (temporizador T2) si se ha recibido alguna primitiva; en caso de ser así, se descodifica y se envía hacia el bloque `MAC_CCF_FT`. Si se recibe una señal de dicho bloque se procede a insertar los datos que transporta en la cola de salida después de codificarlos.

SPICUM servicio de publicaciones

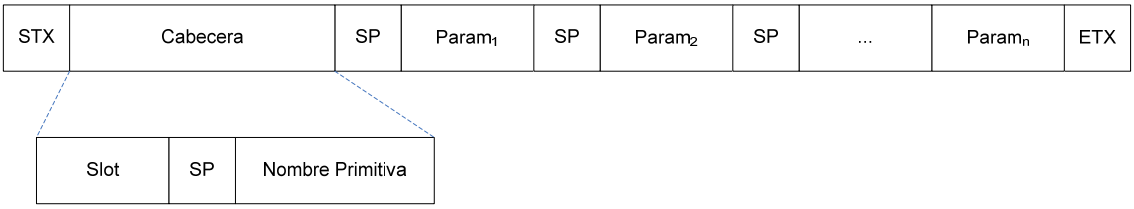
El formato de codificación de las primitivas es distinto según el sentido de la comunicación (Figura J.6). Las tramas descendentes se codifican en hexadecimal, con una cabecera que incluye, entre otra información, el código de la primitiva y la longitud de la misma; tras la cabecera van los parámetros de la primitiva. En el sentido ascendente, cada campo de la primitiva se separa del siguiente por un carácter espacio (`SP`) y en la cabecera se incluye el nombre ASCII de la primitiva. Cada octeto de un parámetro, en sentido ascendente, se codifica como dos octetos que llevan el código ASCII correspondiente al dígito representado en los cuatro bits superiores (primer octeto) y los cuatro bits inferiores (segundo octeto). Las tramas van delimitadas por los caracteres `STX` (*Start of TeXt*) y `ETX` (*End of TeXt*). Los caracteres de control que aparezcan dentro de la trama se escapan con el carácter `DEL` (`0x10`).



Ejemplo: `DBC_REQ 3 : 1`

Codificado: `2: 0: 5: 0: -1: 10: 2: 10: 3: 1: 3`

(a)



Ejemplo: `03 CSF_TBC_IND 00 40 1E 23 45`

Codificado:
```
2 30 33 20 43 53 46 5F 54 42 43 5F 49 4E 44 20 20 20 20 20
  20 30 30 20 34 30 20 31 45 20 32 33 20 34 35 20 20  A  3
```

(b)

Figura J.6: Formato y ejemplos de las tramas (a) descendentes y (b) ascendentes de la Interfaz con el Módulo de Capa Física.

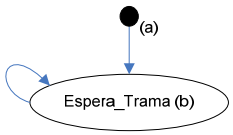
El diagrama de estados de este proceso y las transiciones posibles en cada uno de ellos para las señales válidas se muestran en la Figura J.7. El comportamiento de cada estado es el siguiente:

- `Espera_Trama`: Es el único estado del proceso. Se llega a él tras la inicialización del Módulo de Capa Física, y se encarga de la recepción y envío de señales a través de las colas creadas para entrada y salida, respectivamente. Si hay un error de transmisión, se reintenta el envío varias veces; si sigue fallando, se informa de dicha circunstancia (`CSF_LINSER_ERR`). Cuando vence el temporizador `T2` se comprueba si existe algún mensaje a la entrada.

SPICUM servicio de publicaciones

Señales	Estados a	Estados b
-	b	
CSF_CO_DATA_REQ		b
CSF_DBC_REL_REQ		b
CSF_DBC_REQ		b
CSF_INIT_REQ		b
CSF_LINSER_RESET		b
CSF_PAGE_INFO_FULL_REQ		b
CSF_PAGE_INFO_SHORT_REQ		b
CSF_SET_HO_REQ		b
CSF_TBC_BUF_REQ		b
CSF_TBC_REL_REQ		b
CSF_TBC_RES_REQ		b
T2		b

(a)



(b)

Figura J.7: (a) Transiciones posibles y (b) Diagrama de estados para el proceso LINSER de la Terminación Fija.

J.1.2 Nivel de Control del Enlace

En este apartado se describe el comportamiento de los procesos que integran los bloques cuya funcionalidad se corresponde con el Nivel de Control del Enlace, así como los estados que posee cada uno de ellos. La lista de estos procesos, agrupados por el bloque al que pertenecen, se muestra en la Tabla J.2.

Tabla J.2: Lista de procesos que, funcionalmente, se encuentran asociados al Nivel DLC.

Bloque	Procesos
DLC_FT	CTRL_FT
	LAPC_FT
	Lc_FT
	SignalROUTER
	Lb_FT
SUB_DLC_FT	ConversorTTCN
	Cuasi_Lc
	Signal_RTX
AJUSTE_TIPOS_FT	AJUSTE_TIPOS_FT

J.1.2.1 Proceso CTRL_FT

El proceso CTRL_FT se encarga del control de las conexiones. Su principal función es la creación de entidades LAPC y Lc, así como su gestión posterior. La otra función que realiza es el encaminamiento de las primitivas de liberación de conexión (MAC_DIS_REQ, MAC_DIS_IND) entre el Nivel de Gestión y el Nivel MAC.

El diagrama de estados de este proceso y las transiciones posibles para las señales válidas se muestran en la Figura J.8. El comportamiento del único estado que existe es el siguiente:

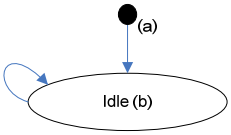
- Idle: Al recibir una indicación de nueva conexión (MAC_CON_IND) se crea una pareja de entidades LAPC-Lc y se enlazan; en el caso de un traspaso simplemente

SPICUM servicio de publicaciones

se informa de la nueva entidad involucrada en la conexión. Si se recibe una petición o una indicación de desconexión, se retransmite a la entidad adecuada (`LLME_FT` o `MAC_CCF_FT`).

Señales	Estados a	b
-	b	
Lc_MAC_DIS_REQ		b
Lc_MOD		b
MAC_CON_IND		b
MAC_DIS_IND		b

(a)



(b)

Figura J.8: (a) Transiciones posibles y (b) Diagrama de estados para el proceso `CTRL_FT` de la Terminación Fija.

J.1.2.2 Proceso LAPC_FT

El proceso `LAPC_FT` proporciona un servicio de transferencia bidireccional confirmada de ventana uno con retransmisiones. Entre sus funciones se encuentran el control de flujo y la detección y recuperación de errores. La Terminación Fija espera solicitudes de conexión de una Terminación Portátil, que es siempre la iniciadora en el Perfil GAP.

Para realizar la comunicación se emplean dos tipos de tramas. Las tramas `I` (*Information*) transportan datos del Nivel de Red; por su parte, las tramas `RR` (*Receive Ready*) representan las confirmaciones e indican la disponibilidad para recibir tramas `I` adicionales. Ambos extremos mantienen las variables típicas asociadas a una transmisión con confirmación:

- `V(S)`: Número de secuencia de la próxima trama `I` a transmitir.
- `V(A)`: Número de secuencia de la última trama `I` confirmada.
- `V(R)`: Número de secuencia esperado de la próxima trama `I` a recibir.

Igualmente, las tramas incluyen también la información típica:

- `N(S)`: Número de secuencia de la trama. Este valor sólo se incluye en tramas `I`.
- `N(R)`: Número de secuencia esperado de la próxima trama `I` a recibir. Sirve como confirmación de todas las tramas anteriores (hasta `N(R)-1`). I. Sólo son válidos valores que cumplan la condición `V(A)`≤`N(R)`≤`V(S)`. Este valor se incluye en tramas `I` y tramas `RR`.

La transmisión de información (`DL_DATA_REQ`) entre ambos extremos se realiza como se indica en la Figura J.9; en el establecimiento (`DL_ESTABLISH_REQ`) tanto `X` como `Y` son cero, con el bit `NLF` (*New Link Flag*) a 1 (en el resto de las tramas se pone a 0). Las pérdidas de tramas y los posibles errores de recepción se controlan con el temporizador `DL.04`, que se activa al transmitir una trama y se desactiva al recibir su confirmación; si vence este temporizador, se retransmite hasta un máximo de tres ocasiones (contador `N250`). El restablecimiento del enlace se puede realizar en cualquier momento siguiendo el procedimiento normal de establecimiento; si esto sucede, se descartan todas las tramas y datos en curso y se inicializan las variables de estado.

En el sentido de transmisión, se ha modelado una cola que almacena todas las PDUs de Nivel de Red cuya transmisión ha sido solicitada. Cuando se recibe una indicación de

SPICUM servicio de publicaciones

que es posible enviar más datos o se confirma la última trama enviada se construye una nueva trama `I` que contiene la primera PDU de esta cola. En recepción, las tramas recibidas ya han sido recombinadas por la entidad `Lc`, por lo que se reenvían al Nivel de Red.

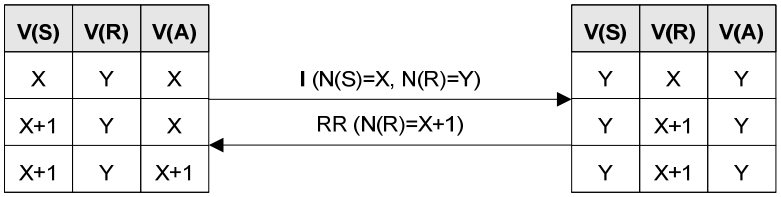


Figura J.9: Transmisión de información con confirmación.

Esta entidad es también responsable de codificar y descodificar la información del Nivel de Red. Esto es así porque la interfaz con el Subsistema de Pruebas transporta esta información descodificada. Antes de incluir un mensaje del Nivel de Red en la cola de transmisión, se codifica invocando al procedimiento `Conv_Octet_PDU_NWK`; de igual forma, en recepción, antes de reenviar el mensaje al Nivel de Red, se descodifica invocando al procedimiento `Conv_Primi_PDU_NWK`. La implementación de estas funciones está descrita en la Sección J.3.1.

El diagrama de estados de este proceso y las transiciones posibles en cada uno de ellos para las señales válidas se muestran en la Figura J.10. El comportamiento de cada estado es el siguiente:

- `Idle`: Es el estado inicial. El proceso acaba de ser creado y está a la espera de la notificación del Nivel de Gestión (`LAPC_CON_IND`) que le indique los parámetros del enlace.
- `Wait_Dlc_Cnx`: Se espera la trama de establecimiento del enlace (primera trama `I`). Cuando llega, se confirma, se avisa al Nivel de Red (`DL_ESTABLISH_IND`) [1] y se pasa al estado `Connected`.
- `Connected`: Este estado gestiona la transferencia de datos de un enlace ya establecido, así como las liberaciones y los restablecimientos del enlace[2].

  Las peticiones de transmisión del Nivel de Red (`DL_DATA_REQ`) se codifican y se colocan en la cola de transmisión; en tanto un mensaje no sea confirmado, no se elimina de esta cola. La transmisión se realiza en orden de petición y puede venir provocada por una confirmación (si hay mensajes a transmitir se transmite el primero) o por el vencimiento del temporizador `DL.04` (se retransmite la última trama enviada, ya que no ha sido confirmada). El campo de la suma de comprobación se pone a cero en esta entidad, ya que se calcula en la entidad `Lc_FT` asociada. En recepción (`LC_TRAMA_IND`), se comprueba la validez de la trama recibida, se descodifican los datos, se confirma la recepción con una trama

[1] En la tramas de la fase de establecimiento pueden incluirse datos sólo en el caso del mensaje `PDU_PAGE_RESPONSE`.

[2] Las peticiones de restablecimiento se tratan iniciando el procedimiento normal de establecimiento del enlace.

SPICUM servicio de publicaciones

RR (o una trama I si quedan datos por enviar) y se elimina el mensaje confirmado de la cola de transmisión.

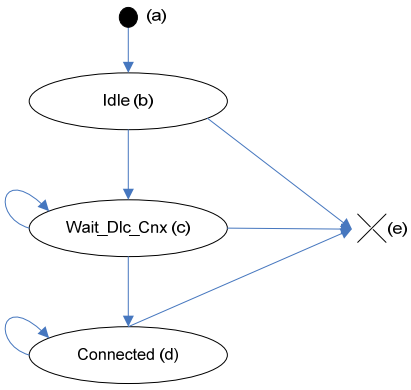
La liberación del enlace puede ser solicitada por el Nivel de Red (DL_RELEASE_REQ) o por el otro extremo (LAPC_DIS_IND), y puede ser de tipo normal o anormal. Se considera una liberación anormal si se indica así explícitamente o si lo solicita el otro extremo habiendo aún datos por transmitir. En este caso se descartan los datos aún no transmitidos. En las liberaciones normales no se cierra el enlace hasta que se han transmitido todos los datos pendientes. En ambos casos se avisa al Nivel de Gestión para que libere los recursos asignados a este enlace en el Nivel MAC. Finalmente, se destruye la instancia del proceso LAPC_FT asociada al enlace.

Este proceso incluye los siguientes procedimientos relevantes:

- ProcesarTramaRX_FT: Comprueba el tipo de trama recibida, si es válida o no, cancela el temporizador de retransmisión (DL.04) y actualiza las variables de estado. Devuelve un código indicando la acción a realizar en respuesta a la trama recibida.
- LiberarCnx_FT: Implementa la liberación del enlace, notificando a la entidad adecuada según el origen de la petición.

	Estados				
Señales	a	b	c	d	e
-	b				
DL.04				d,e	
DL_DATA_REQ				d,e	
DL_RELEASE_REQ			d,e	d,e	
LAPC_CON_IND		c			
LAPC_DIS_IND		e	e	e	
Lb_TRAMA_IND			d,c	d,e	

(a)



(b)

Figura J.10: (a) Transiciones posibles y (b) Diagrama de estados para el proceso LAPC_FT de la Terminación Fija.

J.1.2.3 Proceso Lc_FT

El proceso Lc_FT es el encargado de fragmentar y recombinar las tramas DLC y controlar y calcular la suma de comprobación en la comunicación entre el Nivel DLC y el Nivel MAC. Cuando la entidad CTRL_FT recibe la petición de una nueva conexión, se crea una nueva instancia de este proceso, asociándose a una instancia de la entidad LAPC.

En recepción se dispone de una cola donde se almacenan los fragmentos recibidos. Cuando se recibe (MAC_CO_DATA_IND) un nuevo fragmento (de tamaño 5 octetos en el Perfil GAP) hay que determinar si es el primero de una trama DLC o no. Para ello, se trata como si fuera el primero y se calcula, a partir de la información de la cabecera, cuántos fragmentos componen la trama. Al recibir todos los fragmentos necesarios se

SPICUM servicio de publicaciones

verifica la suma de comprobación; si es correcta, se recombina la trama DLC y se pasa al Nivel DLC, y si no lo es, se descarta el primer fragmento y se vuelve a iniciar el algoritmo con el siguiente fragmento (el primero que haya en la cola de recepción tras el fragmento descartado)[3].

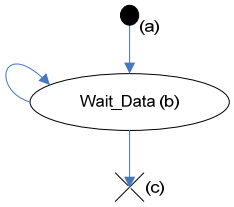
En transmisión, tras calcular la suma de comprobación, las tramas DLC se fragmentan en unidades de cinco octetos y se van transmitiendo (`MAC_CO_DATA_REQ`) cuando el Nivel MAC está dispuesto a aceptar un nuevo fragmento (`MAC_CO_DTR_IND`). Los fragmentos ya enviados son eliminados de la cola de transmisión pues la recuperación de errores se realiza en la entidad `LAPC`.

El diagrama de estados de este proceso y las transiciones posibles para las señales válidas se muestran en la Figura J.11. El comportamiento del único estado que existe es el siguiente:

- `Wait_Data`: En este estado se responde a las peticiones de transmisión de una trama DLC, a las indicaciones de recepción de un nuevo fragmento y a las indicaciones de que se puede transmitir un nuevo fragmento.

	Estados		
Señales	a	b	c
-	b		
Lc_MOD		b, c	
Lc_TRAMA_REQ		b	
MAC_CO_DATA_IND		b	
MAC_CO_DTR_IND		b	

(a)



(b)

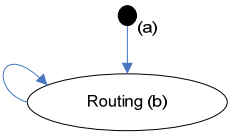
Figura J.11: (a) Transiciones posibles y (b) Diagrama de estados para el proceso `Lc_FT` de la Terminación Fija.

J.1.2.4 Proceso `SignalROUTER`

El proceso `SignalROUTER` es el responsable de encaminar las señales recibidas, tanto del Nivel de Red como del Nivel MAC, a la instancia adecuada para su tratamiento. Las señales del Nivel de Red (`DL_DATA_REQ`, `DL_RELEASE_REQ`) se encaminan a instancias de la entidad `LAPC_FT` y las señales del Nivel MAC (`MAC_CO_DATA_IND`, `MAC_CO_DTR_IND`) se encaminan a instancias de la entidad `Lc_FT`.

	Estados	
Señales	a	b
-	b	
DL_DATA_REQ		b
DL_RELEASE_REQ		b
MAC_CO_DATA_IND		b
MAC_CO_DTR_IND		b

(a)



(b)

Figura J.12: (a) Transiciones posibles y (b) Diagrama de estados para el proceso `SignalRouter` de la Terminación Fija.

[3] En este algoritmo hay que considerar que en el Perfil GAP la longitud máxima de la trama DLC son 70 octetos.

SPICUM servicio de publicaciones

El diagrama de estados de este proceso y las transiciones posibles para las señales válidas se muestran en la Figura J.12. El comportamiento del único estado que existe es el siguiente:

- `Routing`: El encaminamiento de las señales del Nivel de Red y del Nivel MAC se hace, respectivamente, en base al identificador de la conexión a Nivel DLC (`dlei`) y a Nivel MAC (`mcei`).

J.1.2.5 Proceso `Lb_FT`

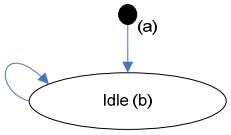
El proceso `Lb_FT` es el encargado de trasladar al Nivel MAC la información de difusión cuya transmisión solicita el Nivel de Red.

El diagrama de estados de este proceso y las transiciones posibles para las señales válidas se muestran en la Figura J.13. El comportamiento del único estado que existe es el siguiente:

- `Idle`: Reenvía la petición de información de difusión (`DL_BROADCAST_REQ`) del Nivel de Red al Nivel MAC (`MAC_PAGE_REQ`).

	Estados	
Señales	a	b
-	b	
DL_BROADCAST_REQ		b

(a)



(b)

Figura J.13: (a) Transiciones posibles y (b) Diagrama de estados para el proceso `Lb_FT` de la Terminación Fija.

J.1.2.6 Proceso `ConversorTTCN`

El proceso `ConversorTTCN` pertenece al bloque de adaptación de señales entre el Subsistema de Pruebas y el Módulo de Protocolos. Realiza la conversión entre las primitivas utilizadas en el Juego de Pruebas (`MAC_DATA_REQ`, `MAC_DATA_IND`), donde la información del Nivel superior viene sin codificar, y las primitivas empleadas en la interfaz superior del Nivel MAC (`MAC_CO_DATA_REQ`, `MAC_CO_DATA_IND`), donde la información del Nivel superior se representa ya codificada.

Al recibir una primitiva, se invoca a la función de codificación o descodificación según se reciba del Nivel DLC (`Conv_Primi_PDU_DLC`) o del Nivel MAC (`Conv_Octet_PDU_DLC`), respectivamente (ver Sección J.3.2);. En el caso de una recepción, se descodifica la cabecera; si es una trama de Información, se invoca al procedimiento `Conv_Octet_PDU_DATA_DLC`, que identifica el mensaje del Nivel de Red transportado por la primitiva y lo descodifica. En transmisión, codifica la trama según si es de Información o de Confirmación; si es de Información, la función `ConvDLCaNWK_SD` codifica los datos del Nivel de Red. El diseño de estas funciones sigue la misma filosofía que las funciones de codificación y descodificación de las PDUs de Nivel de Red. (Sección J.3.1)

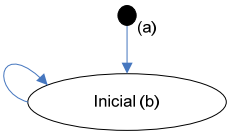
El diagrama de estados de este proceso y las transiciones posibles para las señales válidas se muestran en la Figura J.14. El comportamiento del único estado que existe es el siguiente:

SPICUM servicio de publicaciones

- `Inicial`: Realiza la codificación (`DATA_IN`) o descodificación (`MAC_DATA_REQ`), según corresponda, y reenvía el resultado.

	Estados	
Señales	a	b
-	b	
DATA_IN		b
MAC_DATA_REQ		b

(a)



(b)

Figura J.14: (a) Transiciones posibles y (b) Diagrama de estados para el proceso `ConversorTTCN` de la Terminación Fija.

J.1.2.7 Proceso `Cuasi_Lc`

El proceso `Cuasi_Lc` pertenece al bloque de adaptación de señales entre el Subsistema de Pruebas y el Módulo de Protocolos. Realiza unas funciones similares a las de la entidad `Lc_FT`, fragmentando y recombinando las tramas DLC, verificando la suma de comprobación y gestionando el control de flujo MAC. En recepción, mantiene una cola de fragmentos recibidos; cuando se completa una trama DLC, la cede al proceso `ConversorTTCN`.

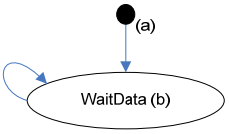
Al arrancar, inicializa las colas de fragmentos de transmisión y recepción y pasa al estado `WAIT_DATA`. Al recibir (`DATA_OUT`) una trama del nivel superior (del Subsistema de Pruebas), la fragmenta e inserta estos fragmentos en la cola de transmisión; estos fragmentos se irán transmitiendo con cada petición de nuevos datos por parte del Nivel MAC (`MAC_CO_DTR_IND`). Por su parte, cada nuevo fragmento entregado por el Nivel MAC (`MAC_CO_DATA_IND`) se inserta en la cola de recepción y se comprueba si se dispone ya de un mensaje completo. Si la longitud de la cola es mayor de 70 (límite de la trama DLC en GAP), se descarta el fragmento más antiguo; lo mismo sucede si la verificación de la suma de comprobación falla.

El diagrama de estados de este proceso y las transiciones posibles para las señales válidas se muestran en la Figura J.15. El comportamiento del único estado que existe es el siguiente:

- `WaitData`: Gestiona las peticiones de transmisión de datos (`DATA_OUT`), las indicaciones de recepción (`MAC_CO_DATA_IND`) y el control de flujo (`MAC_CO_DTR_IND`). Cuando hay una trama DLC completa, la reenvía (`DATA_IN`).

	Estados	
Señales	a	b
-	b	
DATA_OUT		b
MAC_CO_DATA_IND		b
MAC_CO_DTR_IND		b

(a)



(b)

Figura J.15: (a) Transiciones posibles y (b) Diagrama de estados para el proceso `Cuasi_Lc` de la Terminación Fija.

SPICUM servicio de publicaciones

J.1.2.8 Proceso `Signal_RTX`

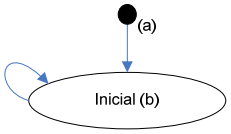
El proceso `Signal_RTX` pertenece al bloque de adaptación de señales entre el Subsistema de Pruebas y el Módulo de Protocolos. Reenvía las señales que no requieren modificación entre ambos Módulos; estas señales son la mayoría, con la excepción de las señales de petición y notificación de datos.

El diagrama de estados de este proceso y las transiciones posibles para las señales válidas se muestran en la Figura J.16. El comportamiento del único estado que existe es el siguiente:

- `Inicial`: Reenvía las señales recibidas.

	Estados	
Señales	a	b
-	b	
MAC_CON_IND		b
MAC_DIS_IND		b
MAC_DIS_REQ		b
MAC_PAGE_REQ		b

(a)



(b)

Figura J.16: (a) Transiciones posibles y (b) Diagrama de estados para el bloque `Signal_RTX` de la Terminación Fija.

J.1.2.9 Proceso `AJUSTE_TIPOS_FT`

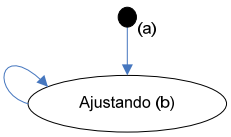
El proceso `AJUSTE_TIPOS_FT` adapta los mensajes `PDU_IDENTITY_REPLY` y `PDU_AUTH_REJECT` de los Juegos de Pruebas para poder emplear internamente una misma señal para ambas Terminaciones. El resto de los mensajes simplemente los reenvía.

El diagrama de estados de este proceso y las transiciones posibles para las señales válidas se muestran en la Figura J.17. El comportamiento del único estado que existe es el siguiente:

- `Ajustando`: Realiza la adaptación de los mensajes `PDU_IDENTITY_REPLY` y `PDU_AUTH_REJECT`, reenviando los demás sin modificaciones.

	Estados	
Señales	a	b
-	b	
DL_BROADCAST_REQ		b
DL_DATA_IND		b
DL_DATA_REQ		b
DL_ESTABLISH_IND		b
DL_RELEASE_CFM		b
DL_RELEASE_IND		b
DL_RELEASE_REQ		b

(a)



(b)

Figura J.17: (a) Transiciones posibles y (b) Diagrama de estados para el proceso `AJUSTE_TIPOS_FT` de la Terminación Fija.

SPICUM servicio de publicaciones

J.1.3 Nivel de Gestión

El Nivel de Gestión de la Terminación Fija está constituido por dos procesos, `LLME_MAC_FT` y `LLME_DLC_FT`, éste último sólo presente en el Sistema de Prueba para el Nivel de Red. Cada uno de estos procesos se encarga de la gestión de un Nivel, MAC o DLC, respectivamente. A continuación se describe el comportamiento de cada uno de los procesos y sus estados.

J.1.3.1 Proceso `LLME_MAC_FT`

El proceso `LLME_MAC_FT` es el responsable de decidir si es posible crear una nueva conexión o realizar un traspaso; para ello, almacena información sobre todas las conexiones activas como, por ejemplo, los canales físicos en uso. Además, proporciona al Nivel MAC la información de configuración de la Terminación Fija que debe difundir (`MAC_ME_CHAN_INFO_REQ`[4]).

Al recibir una petición de conexión (`CSF_ACCESS_REQ`), el Nivel de Gestión, si lo autoriza, le asigna un nuevo identificador `mcei`; el resultado se comunica al Nivel MAC. Si la petición se refiere a un traspaso, se asigna un nuevo identificador `mcei` en el caso de un traspaso de conexión, pero no en el de traspaso de portadora. En este segundo caso, sin embargo, se activa el temporizador `T_HOVER` que determina el tiempo que pueden coexistir abiertas ambas portadoras; este temporizador se detiene cuando el Nivel MAC informa del final del procedimiento de traspaso (`FIN_HOVER`). Cuando se cierra una conexión, actualiza la información sobre recursos ocupados.

Parte de la funcionalidad de este proceso está relacionada con la configuración del comportamiento del Nivel MAC y la consulta de dicha configuración, tal y como es necesario para la ejecución de las Pruebas. Se puede configurar la identidad, los derechos de acceso y las capacidades de la Terminación Fija (`CONFIG_FT`), consultar y modificar la información de capacidades de la Terminación Fija (`TTCN_MAC_FPCAP_IND`, `TTCN_MAC_FPCAP_REQ`), cambiar la identidad del sistema (`TTCN_MAC_RFPI_REQ`), activar la autorización de traspasos (`TTCN_MAC_CHO_PERM_REQ`, `TTCN_MAC_BHO_PERM_REQ`) y solicitar la difusión de la información de los canales no disponibles (`TTCN_MAC_TX_BLIND_SLOT_REQ`).

Este proceso también se encarga de provocar la reinicialización del Nivel MAC por indicación expresa de las Pruebas (`RESET_FT`) o por una pérdida de sincronismo (`MAC_ME_SINC_ERR_IND`).

El diagrama de estados de este proceso y las transiciones posibles en cada uno de ellos para las señales válidas se muestran en la Figura J.18. El comportamiento de cada estado es el siguiente:

- `Espera_Conex`: Es el estado principal, en el que se responde a todas las peticiones, tanto de creación o traspaso de conexiones como de configuración del Nivel MAC.
- `Espera_Reset_Ind`: Se espera en este estado a que el Nivel MAC cierre las conexiones activas tras la notificación de una reinicialización.

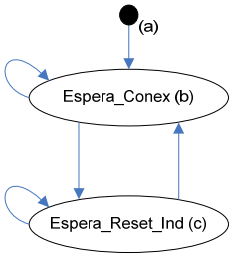
En ambos estados se pueden recibir solicitudes acerca de la información de difusión a transmitir.

[4] El tipo de información solicitada se indica en un parámetro de esta primitiva.

SPICUM servicio de publicaciones

	Estados		
Señales	a	b	c
-	b		
CONFIG_FT		c	
FIN_HOVER		b	c
MAC_ME_CHAN_INFO_REQ		b	c
MAC_ME_CON_IND		b	
MAC_ME_DIS_IND		b	c
MAC_ME_SINC_ERR_IND		b	b
RESET_FT		c	
T_HOVER		b	c
TTCN_MAC_BHO_PERM_REQ		b	
TTCN_MAC_CHO_PERM_REQ		b	
TTCN_MAC_FPCAP_IND		b	
TTCN_MAC_FPCAP_REQ		b,c	
TTCN_MAC_RFPI_REQ		b,c	
TTCN_MAC_TX_BLIND_SLOT_REQ		b	

(a)



(b)

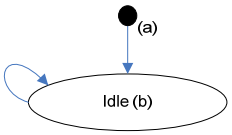
Figura J.18: (a) Transiciones posibles y (b) Diagrama de estados para el proceso `LLME_MAC_FT` de la Terminación Fija.

J.1.3.2 Proceso `LLME_DLC_FT`

El proceso `LLME_DLC_FT` gestiona las conexiones MAC[5], almacena y proporciona los parámetros de identidad (`pmid`, `mcei`) y controla el traspaso de las conexiones[6]. Para cada conexión se almacenan las identidades de las Terminaciones involucradas y los identificadores de las instancias `Lc_FT` y `LAPC_FT` asociadas; la lista de las conexiones activas, que contiene esta información, se exporta para que sea visible en los demás procesos.

	Estados	
Señales	a	b
-	b	
LAPC_DIS_REQ		b
Lc_MAC_CON_IND		b
Lc_MAC_DIS_IND		b
LLME_DLC_LINKPRESENT_REQ		b

(a)



(b)

Figura J.19: (a) Transiciones posibles y (b) Diagrama de estados para el proceso `LLME_DLC_FT` de la Terminación Fija.

En las peticiones de conexión, se crea un nuevo registro de conexión y se comunica a la nueva entidad `LAPC_FT`; si es un traspaso, se asocia la nueva conexión al enlace de forma inmediata. La liberación puede venir indicada desde los procesos `LAPC_FT` (iniciada por el Nivel de Red) y `CTRL_FT` (iniciada por el Nivel MAC); en ambos casos, se informa a las instancias asociadas a la conexión y se elimina el correspondiente registro de conexión.

[5] Puede gestionar simultáneamente múltiples conexiones.

[6] La creación y traspaso de una conexión son procedimientos controlados por la Terminación Portátil. La Terminación Fija se limita a esperar dichas indicaciones y responder adecuadamente.

SPICUM servicio de publicaciones

El diagrama de estados de este proceso y las transiciones posibles para las señales válidas se muestran en la Figura J.19. El comportamiento del único estado que existe es el siguiente:

- `Idle`: Responde a las peticiones de creación, traspaso y liberación de una conexión.

J.2 Sistema de Pruebas para la Terminación Fija

En esta Sección se describe el modelado de los procesos que integran el Módulo de Protocolos para el Sistema de Pruebas de la Terminación Fija, `SP_DLC_FT`. La lista de estos procesos, y los bloques a que pertenecen, se muestra en la Tabla J.1. En primer lugar se describen los procesos relacionados con el Nivel de Acceso al Medio, después los relacionados con el Nivel de Control del Enlace y, finalmente, los relacionados con el Nivel de Gestión.

La descripción que se presenta en esta Sección hace referencia tan sólo a las diferencias de cada uno de los procesos con respecto a su homólogo en los Sistemas de Pruebas para Terminaciones Portátiles descritos en la Sección anterior.

J.2.1 Nivel de Acceso al Medio

El Nivel de Acceso al Medio tiene un comportamiento similar al de los Sistemas de Prueba para la Terminación Portátil, siendo sus principales diferencias el flujo de la información de difusión (ascendente en este caso) y el control de la creación de las conexiones de tráfico. Al igual que anteriormente, también se ha incluido aquí la descripción del proceso del bloque `LINSER_FT` por considerar que, funcionalmente, está asociado al Nivel MAC.

J.2.1.1 Proceso `BMC`

El proceso `BMC` de la Terminación Portátil realiza la funcionalidad inversa al correspondiente proceso de la Terminación Fija, ya que mientras este último se responsabilizaba de la difusión información, en la Terminación Portátil se debe encargar de recibir la información difundida y notificarla al Nivel de Gestión, que la almacenará para su uso futuro. Por este motivo el diagrama de estados es considerablemente distinto.

El diagrama de estados de este proceso y las transiciones posibles en cada uno de ellos para las señales válidas se muestran en la Figura J.18. El comportamiento de cada estado es el siguiente:

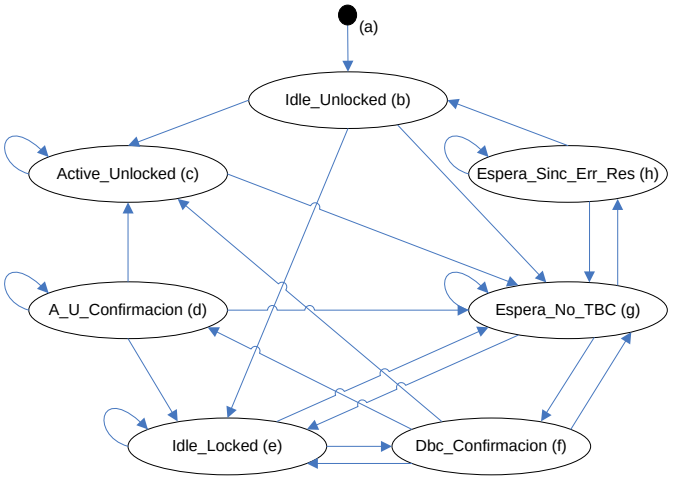
- `Idle_Unlocked`: Es el estado inicial en el que se sitúa el Nivel MAC tras su creación. Espera una indicación de que el Módulo de Capa Física se ha inicializado correctamente, la cual dispara la transición al estado `Active_Unlocked`.
- `Active_Unlocked`: En este estado se escanean periódicamente (`T_SCAN`) las frecuencias de las portadoras; se considera que ha habido una detección correcta cuando se completa la primera fase de la secuencia de enganche (recepción de las señales `CSF_NT_IND`, `CSF_STAT_INFO_IND` y `CSF_FP_CAP_IND`). En este momento se crea un controlador de canal de difusión y se pasa al estado `A_U_Confirmacion`.

SPICUM servicio de publicaciones

- `A_U_Confirmacion`: Espera a recibir las mismas tres señales que en la primera fase de la secuencia de enganche, con los mismos valores. Si esto ocurre, se notifica la información recibida al Nivel de Gestión (ej: identidad `RFPI`) y se le informa de que ya puede permitir la creación de conexiones de tráfico; si se produce algún fallo se continúa la búsqueda de Terminaciones Fijas en el estado anterior.

	Estados							
Señales	a	b	c	d	e	f	g	h
-	b							
CSF_FP_CAP_IND		e		d,c,e		d,c	f,e	
CSF_LINSER_ERROR		g	g	g	g	g	g	g
CSF_LINSER_START		c						
CSF_NT_IND		e		d,c,e		d,c	f,e	
CSF_PAGE_SHORT_IND					e			
CSF_RSSI_TABLE					e			
CSF_STAT_INFO_IND		e		d,c,e		d,c	f,e	
CSF_TIMEOUT_IND					g			
MAC_ME_MAD_HO_IND		g	g	g	g	g	g	g
MAC_ME_RESET_PT		g	g	g	g	g	g	g
MAC_ME_SINC_ERR_RES								b
MBC_BMC_NO_TBC							h	
MBC_CTRL_DBC_REQ					f		g	h
T_SCAN			c	c		e		

(a)



(b)

Figura J.20: (a) Transiciones posibles y (b) Diagrama de estados para el proceso `BMC` de la Terminación Portátil.

- `Idle_Locked`: Se llega a este estado cuando la Terminación Portátil se ha sincronizado con una Terminación Fija. A partir de este punto se está en disposición de recibir mensajes de difusión, generales o de búsqueda. La información que se recibe por difusión se comunica al Nivel de Gestión.
- `Dbc_Confirmacion`: Es un estado similar a `A_U_Confirmacion`. Espera a recibir la secuencia de mensajes que confirma la activación del canal de tráfico. Si tiene éxito informa al controlador de tráfico (proceso `MBC_CTRL`) con `BMC_CBC_CFM`.
- `Espera_No_TBC`: Estado en el que, durante la fase de cierre, se espera la confirmación de que todas las conexiones abiertas se han cerrado

SPICUM servicio de publicaciones

adecuadamente. La fase de cierre se puede iniciar debido a un error de transmisión (número máximo de intentos de transmisión fallidos), la imposibilidad de realizar un traspaso de portadora o una solicitud de cierre por parte del Nivel de Gestión.

- `Espera_Sinc_Err_Res`: Espera que el Nivel de Gestión informe del canal actual en el que está enganchada la Terminación Portátil, para poder cerrar el correspondiente canal de tráfico. Sólo es necesario en el caso de un traspaso de conexión o portadora[7].

J.2.1.2 Proceso `MBC_CTRL`

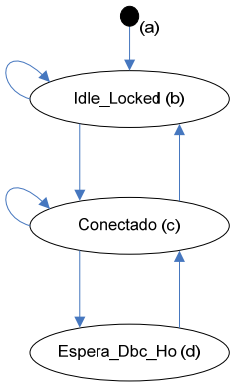
El proceso `MBC_CTRL` se encarga de la creación y mantenimiento de las conexiones de tráfico. En el caso de la Terminación Portátil sólo existe una conexión activa, salvo en caso de que se esté realizando un traspaso. Para almacenar la información de la conexión activa, este proceso utiliza las mismas estructuras de datos que en el caso de la Terminación Fija (Sección J.1.1.2).

El diagrama de estados de este proceso y las transiciones posibles en cada uno de ellos para las señales válidas se muestran en la Figura J.21. El comportamiento de cada estado es el siguiente:

- `Idle_Locked`: La entidad está a la espera de peticiones de conexión. Cuando llega, se solicita el permiso al Nivel de Gestión y, en caso afirmativo, se crea una instancia del proceso `MBC` y se pasa al estado `Conectado`.
- `Conectado`: Enruta los mensajes del Nivel DLC hacia la instancia correspondiente. Responde también a peticiones de desconexión y de traspaso.
- `Espera_Dbc_Ho`: Espera la confirmación de que se puede lanzar una nueva instancia del proceso MBC.

	Estados			
Señales	a	b	c	d
-	b			
BMC_DBC_CFM				c
BMC_MBC_SINC_ERR		b	c	
CSF_HO_IND			c	
MAC_CO_DATA_REQ			c	
MAC_CON_REQ		b,c	d	
MAC_DIS_REQ			c	
MBC_DIS			c,b	

(a)



(b)

Figura J.21: (a) Transiciones posibles y (b) Diagrama de estados para el proceso `MBC_CTRL` de la Terminación Portátil.

[7] En este caso, el canal físico utilizado sólo es conocido por el Nivel de Gestión.

SPICUM servicio de publicaciones

J.2.1.3 Proceso MBC

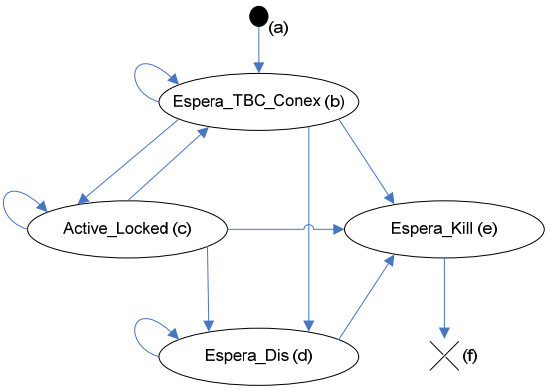
Al igual que en la Terminación Fija (Sección J.1.1.3), las instancias del proceso MBC son creadas dinámicamente cuando se inicia una conexión. Pueden existir hasta dos instancias, una que gestiona la conexión activa y otra que se crea únicamente durante el procedimiento de traspaso. La primera acción a realizar es informar al proceso MBC_SELEC de la nueva instancia, para que encamine adecuadamente los mensajes que reciba. Tras recibir la confirmación del Módulo de Capa Física se considera creada la conexión y se entra en la fase de transferencia de datos.

El diagrama de estados de este proceso y las transiciones posibles en cada uno de ellos para las señales válidas se muestran en la Figura J.21. El comportamiento de cada estado es el siguiente:

- Espera_TBC_Conex: Espera la confirmación de conexión del Módulo de Capa Física (CSF_TBC_CFM), dentro del intervalo indicado por el temporizador T200.
- Active_Locked: Representa el estado de transferencia de información; también gestiona las desconexiones.
- Espera_Dis: Su comportamiento es similar al correspondiente estado de la Terminación Fija.
- Espera_Kill: Su comportamiento es similar al correspondiente estado de la Terminación Fija.

	Estados					
Señales	a	b	c	d	e	f
-	b					
CSF_CO_DATA_IND		b	c	d	e	
CSF_CO_DTR_IND		b	c	d	e	
CSF_DIS_IND		e	e	e	e	
CSF_TBC_CFM		c	c	d	e	
MAC_CO_DATA_REQ		b	c	d	e	
MAC_DIS_REQ		d	d	d	e	
MAC_ENC_EKS_REQ		b	c,d	d	e	
MAC_ENC_KEY_REQ		b	c	d	e	
MBC_KILL		b	c	d	f	
T_200		d	d	d	e	
T_RESP_CSF		e	e	e	e	

(a)



(b)

Figura J.22: (a) Transiciones posibles y (b) Diagrama de estados para el proceso MBC de la Terminación Portátil.

J.2.1.4 Proceso MBC_SELEC

El proceso MBC_SELEC es igual al correspondiente proceso de la Terminación Fija (Sección J.1.1.4); encamina las señales recibidas del Módulo de Capa Física a la instancia adecuada del proceso MBC. Su diagrama de estados y las posibles transiciones están mostrados en la Figura J.5.

J.2.1.5 Proceso LINSER_PT

El proceso LINSER_PT es igual al correspondiente proceso de la Terminación Fija (Sección J.1.1.5); Realiza la comunicación entre el Módulo de Protocolos y el Módulo

SPICUM servicio de publicaciones

de Capa Física. Su diagrama de estados y las posibles transiciones están mostrados en la Figura J.7.

J.2.2 Nivel de Control del Enlace

Los procesos relacionados con el Nivel de Control del Enlace en el Sistema de Pruebas de la Terminación Fija se corresponden con el bloque `SUB_DLC_PT`, que complementa la funcionalidad del Nivel de Control del Enlace no presente en el Juego de Pruebas. La diferencia con los correspondientes procesos de los Sistemas de Prueba para la Terminación Portátil estriba en el proceso `Signal_RTX` y el conjunto de señales que procesa.

J.2.2.1 Proceso `ConversorTTCN`

El proceso `ConversorTTCN` es igual al correspondiente proceso de la Terminación Fija (Sección J.1.2.6); convierte entre las primitivas utilizadas en el Juego de Pruebas (`MAC_DATA_REQ`, `MAC_DATA_IND`) y las primitivas empleadas en la interfaz superior del Nivel MAC (`MAC_CO_DATA_REQ`, `MAC_CO_DATA_IND`). Su diagrama de estados y las posibles transiciones están mostrados en la Figura J.14.

J.2.2.2 Proceso `Cuasi_Lc`

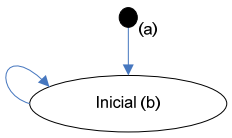
El proceso `Cuasi_Lc` es igual al correspondiente proceso de la Terminación Fija (Sección J.1.2.7); realiza unas funciones similares a las de la entidad `Lc_FT`, fragmentando y recombinando las tramas DLC. Su diagrama de estados y las posibles transiciones están mostrados en la Figura J.15.

J.2.2.3 Proceso `Signal_RTX`

El proceso `Signal_RTX` es similar al correspondiente proceso de la Terminación Fija (Sección J.1.2.8); reenvía las señales que no requieren modificación entre el Subsistema de Pruebas y el Módulo de Protocolos. La diferencia reside en el conjunto de señales que procesa. Su diagrama de estados y las posibles transiciones están mostrados en la Figura J.23.

	Estados	
Señales	a	b
-	b	
MAC_CON_CFM		b
MAC_CON_REQ		b
MAC_DIS_IND		b
MAC_DIS_REQ		b
MAC_PAGE_IND		b

(a)



(b)

Figura J.23: (a) Transiciones posibles y (b) Diagrama de estados para el proceso `Signal_RTX` de la Terminación Portátil.

SPICUM servicio de publicaciones

J.2.3 Nivel de Gestión

El Nivel de Gestión de la Terminación Portátil está constituido por un solo proceso, `LLME_MAC_PT`, correspondiente a la gestión del Nivel MAC. A continuación se describe el comportamiento de este proceso.

J.2.3.1 Proceso `LLME_MAC_PT`

El proceso `LLME_MAC_PT` se encarga, principalmente, de recibir y almacenar la información de difusión procedente de la Terminación Fija y determinar, en función del estado actual, si está permitido establecer una nueva conexión. Además, puede solicitar un reinicialización del sistema, tras una petición externa (`RESET_PT`), y notificar al proceso `BMC` de errores en el Módulo de Capa Física.

El proceso se mantiene en el estado inicial hasta que recibe del proceso `BMC` la indicación de enganche (`MAC_ME_SINC_IND`); el Nivel de Gestión almacena la información de difusión, relativa al canal escuchado, contenida en esta primitiva, y, a partir de este momento, permite la creación de conexiones de tráfico (`MAC_ME_CON_REQ`). La señal `MAC_ME_CON_REQ` se utiliza tanto para solicitar la creación de una conexión como de un traspaso. El tipo de petición puede ser: `CSF_ACCESS_REQ` (petición de conexión), `CSF_BEARER_HO_REQ` (traspaso de portadora) o `CSF_CONN_HO_REQ` (traspaso de conexión).

Las peticiones de traspaso son rechazadas si no existe una conexión activa, al igual que las peticiones de conexión si ya existe una conexión activa (en la Terminación Portátil, en GAP, sólo puede existir una conexión activa). Igualmente, un traspaso se rechaza si está demasiado próximo al último traspaso realizado (`T_HANDOVER`).

En cualquier momento, independientemente del estado en el que se encuentre, el proceso puede recibir peticiones de consulta (`TTCN_MAC_FPCAP_IND`) y actualizaciones de información (`MAC_ME_INFO_IND`, `MAC_ME_CHAN_INFO_IND`), además de indicaciones de desconexión (`MAC_ME_DIS_IND`).

El diagrama de estados de este proceso y las transiciones posibles en cada uno de ellos para las señales válidas se muestran en la Figura J.24. El comportamiento de cada estado es el siguiente:

- `Active_Unlocked`: Es el estado inicial, en el cual la Terminación Portátil no está enganchada a ninguna Terminación Fija. Cuando se recibe la indicación de enganche (`MAC_ME_SINC_IND`) se almacena la información del canal físico a que se ha enganchado y se pasa al estado `Idle_Locked`. Las peticiones de conexión se rechazan en este estado.
- `Idle_Locked`: El Nivel de Gestión se encuentra aquí a la espera de la creación de una conexión de tráfico. Se rechazan aquí las peticiones de traspaso.
- `Conectado`: Se llega a este estado cuando existe una conexión activa. Se aceptan peticiones de traspaso y de desconexión. En el primer caso, se espera la finalización del procedimiento en el estado `Espera_Ho`.
- `Espera_Ho`: Se está a la espera del cierre de una de las dos conexiones activas durante el procedimiento de traspaso. Tras recibir la indicación de desconexión de una de ellas se inicia el temporizador `T_HANDOVER`, que inhibe posibles traspasos durante su duración.

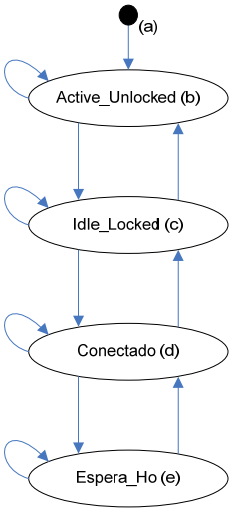
SPICUM servicio de publicaciones

En todos los estados se atiende a actualizaciones de la información de difusión, que son indicadas con las señales:

- `MAC_ME_INFO_IND`: Transporta la información de identidad, derechos de acceso y capacidades de la Terminación Fija
- `MAC_ME_CHAN_INFO_IND`: Incluye la máscara de intervalos no disponibles.

Señales	Estados				
	a	b	c	d	e
-	b				
MAC_ME_B_HO_ERR_IND		b	c	d	e
MAC_ME_CHAN_INFO_IND		b	c	d	e
MAC_ME_CON_REQ		b	c,d	c,e	e
MAC_ME_DIS_IND				c	d
MAC_ME_INFO_IND		b	c	d	e
MAC_ME_SINC_ERR_IND		b	b		
MAC_ME_SINC_IND		c			
RESET_PT		b	c	d	e
T_HANDOVER		b	c	d	e
TTCN_MAC_FPCAP_IND		b	c	d	e
TTCN_MAC_FPCAP_REQ		b	c	d	e

(a)



(b)

Figura J.24: (a) Transiciones posibles y (b) Diagrama de estados para el proceso `LLME_MAC_PT` de la Terminación Portátil.

J.3 Paquetes Auxiliares

En esta Sección se describe la implementación de aquellos paquetes auxiliares que contienen código que modela algún comportamiento, es decir, que incluyen tipos de procesos o procedimientos. Estos paquetes son `Comun_DLC`, `Comun_MAC`, `Sub_DLC` y `Esc_Lec_Serie`. La Tabla J.3 muestra en qué Módulo de Protocolos se emplea cada uno de estos paquetes. Una lista de los tipos de procesos y procedimientos de cada uno de estos paquetes, junto con una breve descripción de los mismos, aparece en el Apéndice I.

Tabla J.3: Paquetes utilizados en el diseño de cada Módulo de Protocolos.

	Módulos de Protocolos		
	MProt_DLC_PT	**MProt_DLC_FT**	**MProt_NWK_PT**
Comun_DLC	-	-	✓
Sub_DLC	✓	✓	-
Comun_MAC	✓	✓	✓
Esc_Lec_Serie	✓	✓	✓

SPICUM servicio de publicaciones

Los paquetes que se han definido son los siguientes:

- `Comun_DLC`: Contiene procedimientos utilizados por ambas Terminaciones en el Nivel de Enlace.
- `Sub_DLC`: Adapta algunas primitivas utilizadas entre los Juegos de Pruebas y el Nivel MAC.
- `Comun_MAC`: Contiene procedimientos y parámetros de configuración del Nivel de Acceso al Medio comunes a ambas Terminaciones.
- `Esc_Lec_Serie`: Ofrece procedimientos de conversión de datos para el Módulo de Capa Física.

J.3.1 Paquete `Comun_DLC`

El paquete `Comun_DLC` contiene procedimientos (Apéndice I, Sección I.3.1) empleados en el Nivel de Enlace en ambas Terminaciones para realizar las siguientes funciones:

- Codificar y descodificar mensajes del Nivel de Red: Estas funciones se han generado automáticamente mediante la herramienta *GenCodecAir* (Capítulo 5, Sección 5.5.3). En total hay 109 en cada sentido, de las cuales, aproximadamente, dos quintas partes corresponden a primitivas de datos mientras que el resto corresponden a tipos estructurados. La descripción de la estructura de estas funciones se encuentra en la Sección mencionada anteriormente.
- Codificar tramas DLC a transmitir, incluyendo funciones para fragmentar dichas tramas y para verificar y generar la suma de comprobación.
- Gestionar las colas de transmisión.
- Obtener el identificador de conexión a partir de la identidad de una de las partes.

J.3.2 Paquete `SUB_DLC`

El paquete `SUB_DLC` permite la adaptación entre el Nivel MAC y el Subsistema de Pruebas. Toda su funcionalidad está agrupada en dos tipos de proceso: `ConversorTTCN` y `Cuasi_Lc`. Estos procesos han sido ya descritos anteriormente en las Secciones J.1.2.6 y J.1.2.7.

Este paquete incluye también los procedimientos de codificación y descodificación de las PDUs de Nivel DLC utilizadas por los Juegos de Pruebas de dicho Nivel (`SP_DLC_PT_Gap.mp`, `SP_DLC_FT_Gap.mp`). La función `Conv_Octet_PDU_DLC` codifica las cuatro PDUs del Nivel de Red transportadas en las tramas de Información que transmiten los Juegos de Pruebas, así como las tramas RR. La función `Conv_Primi_PDU_DLC` se encarga de descodificar las PDUs de Nivel de Red transportadas en tramas RR o I; los Juegos de Pruebas pueden recibir tres PDUs del Nivel de Red. Estas funciones se han generado automáticamente mediante la herramienta *GenCodecAir* (Capítulo 5, Sección 5.5.3).

J.3.3 Paquete `Comun_MAC`

El paquete `Comun_MAC` incluye tres procedimientos empleados internamente para buscar una conexión en la lista de conexiones activas a partir de información como el

SPICUM servicio de publicaciones

identificador de la conexión a Nivel DLC (`mcei`), el intervalo empleado por el canal o el PId del proceso MBC correspondiente.

Este paquete incluye también parámetros de configuración del Nivel MAC de ambas Terminaciones. Entre estos parámetros se encuentran la identidad del sistema, valores por defecto del comportamiento del Nivel, los límites permitidos de fallos en los campos A y B de las tramas de Nivel Físico, los intervalos predefinidos para los canales, etc. Además, declara constantes con la duración de los temporizadores utilizados, como por ejemplo el temporizador de establecimiento de conexión, `T200`, o el tiempo entre búsquedas sucesivas de frecuencia, `TSCAN`. El paquete incluye también constantes utilizadas en las primitivas de comunicación con las placas. Una relación de todos los parámetros de configuración y las constantes de los temporizadores, junto con una breve descripción de cada una, se encuentra en [SANC00a].

J.3.4 Paquete `Esc_Lec_Serie`

El paquete `Esc_Lec_Serie` contiene procedimientos básicos de codificación y descodificación de los parámetros de las primitivas utilizadas en la interfaz entre el Módulo de Protocolos y el Módulo de Capa Física. La lista de estos procedimientos se encuentra en el Apéndice I, sección I.3.4. Además, incluye las librerías y estructuras de datos necesarias para esta comunicación. Por ejemplo, una primitiva se modela con la estructura de datos mostrada en la Figura J.25. El campo `cadena` contiene los parámetros de la primitiva en formato codificado (ver Sección J.1.1.5).

```
struct primitiva {
     int primitiva;
     int tipo;
     char cadena[MAXBUFF];
     int leng;
};
```

Figura J.25: Estructura de datos que contiene una primitiva de la interfaz entre el Módulo de Protocolos y el Módulo de Capa Física.

J.4 Bloques de Emuladores

Los bloques de los emuladores incluyen la funcionalidad mínima imprescindible para realizar las pruebas sobre el nivel de interés. La mayor parte de esta funcionalidad consiste en responder automáticamente a las primitivas que recibe, pero también incorporan canales de comunicación con el exterior del sistema de forma que el operador pueda guiar el desarrollo de las pruebas. Los bloques que se han desarrollado han sido los siguientes:

- `EMU_IWU_PT`
- `EMU_NWK_FT`
- `EMU_NWK_PT`
- `EMU_DLC_FT`
- `EMU_DLC_PT`
- `EMU_MAC`

SPICUM servicio de publicaciones

- EMU_PHY

J.4.1 Emulador EMU_IWU_PT

El Nivel de Red es el nivel superior en la arquitectura del sistema DECT. A las aplicaciones que utilizan los servicios que proporciona esta arquitectura se les denomina, de forma genérica, Unidad de Aplicación (IWU – *InterWorking Unit*). Este elemento ofrece la funcionalidad necesaria para transportar la información al usuario final.

El bloque EMU_IWU_PT se ha diseñado teniendo en cuenta las pruebas que se van a ejecutar. Este emulador se comunica con el Nivel de Red, a través de los Puntos de Acceso al Servicio MNCC y MM, y con el entorno. Consta de dos procesos (Figura J.26): EMU_IWU_CC_PT, que emula la coordinación de las funciones de Control de la Llamada, y EMU_IWU_MM_PT, que emula la coordinación de las funciones de Gestión de la Movilidad. Ambos procesos aceptan comandos externos que activan acciones requeridas por las pruebas.

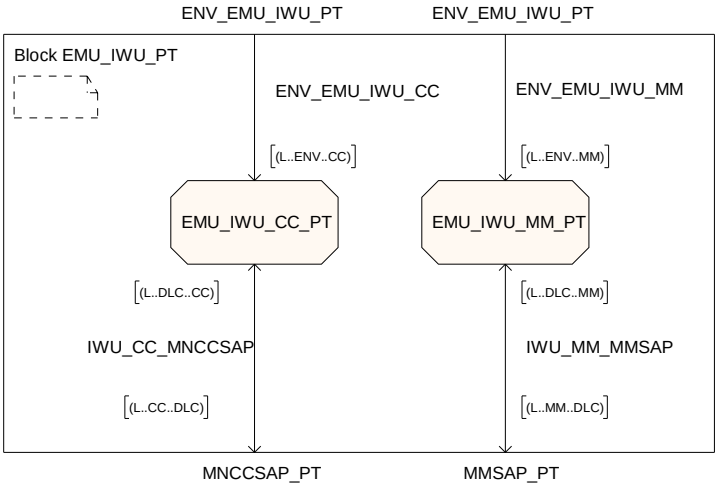


Figura J.26: Modelo del Bloque EMU_IWU_PT.

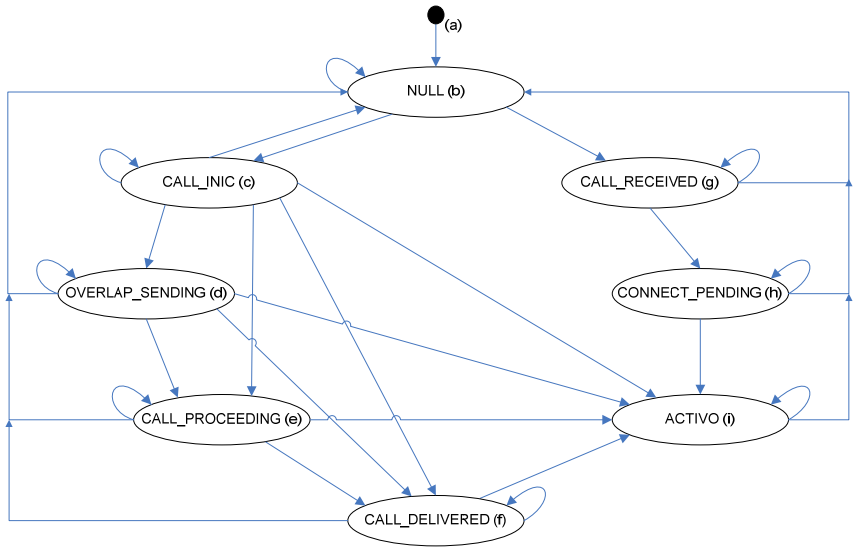
El diagrama de estados del proceso EMU_IWU_CC_PT y las transiciones posibles para las señales válidas se muestran en la Figura J.27. Los estados posibles son los estados principales de la entidad de Control de la Llamada del Nivel de Red. Su comportamiento en el Nivel IWU es el siguiente:

- NULL: Es el estado de reposo. Se alcanza tras cargar los valores adecuados de los parámetros según el GAP y las pruebas que se realizan.
- CALL_INIC, OVERLAP_SENDING, CALL_PROCEEDING, CALL_DELIVERED: Sucesivos estados que pueden alcanzarse durante el establecimiento de una llamada.
- CALL_RECEIVED, CONNECT_PENDING: Estados que ocurren durante el proceso de establecimiento de una llamada entrante.
- ACTIVO: Estado en que la llamada se encuentra activa.

SPICUM servicio de publicaciones

Señales	Estados								
	a	b	c	d	e	f	g	h	i
-	b								
IWU_SETUP		c							
MNCC_SETUP_ACK_IND			d						
MNCC_CALL_PROC_IND			e	e					
MNCC_CONNECT_IND			i	i	i	i			
MNCC_ALERT_IND			f	f	f				
MNCC_SETUP_IND		g							
IWU_CONNECT							h		
MNCC_INFO_IND							g		
MNCC_CONNECT_CFM								i	
MNCC_RELEASE_IND			b	b	b	b	b	b	b
MNCC_RELEASE_CFM			b	b	b	b	b	b	b
MNCC_REJECT_IND			b	b	b	b	b	b	b
IWU_DATOS_X[8]		b	c	d	e	f	g	h	i
IWU_LIBERAR_1			b	b	b	b	b	b	b
IWU_LIBERAR_2[9]			b	b	b	b	b	b	b

(a)



(b)

Figura J.27: (a) Transiciones posibles y (b) Diagrama de estados para el proceso `EMU_IWU_CC_PT` de la Terminación Portátil.

El diagrama de estados del proceso `EMU_IWU_MM_PT` y las transiciones posibles para las señales válidas se muestran en la Figura J.28. El comportamiento de los estados es el siguiente:

- `Reposo`: Es el estado principal, donde se procesan las indicaciones del Nivel de Red y las solicitudes desde el entorno.
- `Espera_LocReq`, `Espera_AuthReq`, `Espera_AccReq` , `Espera_AuthTerReq`: Estados donde se espera la respuesta a la última petición realizada.

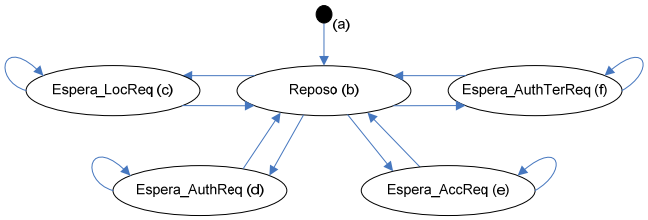
[8] Hay ocho señales que permiten solicitar el envío de diferentes datos.

[9] Se trata de una liberación parcial.

SPICUM servicio de publicaciones

Señales	Estados					
	a	b	c	d	e	f
-	b					
IWU_LOC_REQ		c				
IWU_AUTH_REQ		d				
MM_IDENTITY_IND		b				
MM_AUTHENTICATE_IND		b				
MM_AUTHENTICATE_CFM				b		b
TIMER_IWU			b	b	b	b
MM_LOCATE_CFM			b			
MM_IDENTITY_ASSIGN_IND		b	c	d	e	f
IWU_AR_REQ		e				
MM_ACCESS_RIGHTS_CFM					b	
MM_ACCESS_RIGHTS_TERMINATE_IND		f				
MM_INFO_IND		b				

(a)



(b)

Figura J.28: (a) Transiciones posibles y (b) Diagrama de estados para el proceso `EMU_IWU_MM_PT` de la Terminación Portátil.

J.4.2 Emulador `EMU_NWK_FT`

El emulador del Nivel de Red de la Terminación Fija incorpora la funcionalidad necesaria para poder realizar las pruebas del Juego de Pruebas `SP_DLC_FT_Gap.mp` sobre el Nivel de Enlace de dicha Terminación. Se comunica con el Nivel de Enlace a través de las dos interfaces que ofrece (`SAPB_FT` y `SAP0_FT`) (Figura J.29).

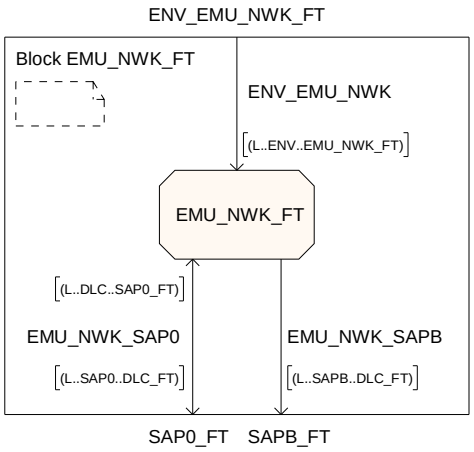


Figura J.29: Modelo del emulador del Nivel de Red de la Terminación Fija.

SPICUM servicio de publicaciones

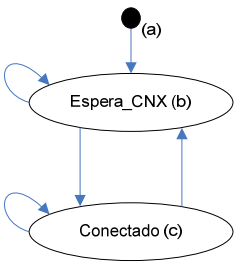
El operador puede solicitar el envío de información de difusión mediante la señal BROADCAST (a través de la ruta ENV_EMU_NWK), permitiendo que la Terminación Portátil inicie el establecimiento del enlace. La información de difusión se rellena de acuerdo con los Parámetros de Pruebas escogidos. Tras inicializarse, el proceso se queda a la espera que la Terminación Portátil establezca un enlace válido. Mientras la conexión está activa, cualquier dato que se reciba en el Nivel de Red es reenviado de vuelta.

El emulador contiene un único proceso. Su diagrama de estados y las transiciones posibles en cada uno de ellos para las señales válidas se muestran en la Figura J.30. El comportamiento de cada estado es el siguiente:

- Espera_CNX: Es el estado inicial, a la espera de recibir la indicación de establecimiento del enlace.
- Conectado: Los datos recibidos a Nivel de Red son copiados y reenviados hacia la Terminación Portátil (DL_DATA_REQ). Cuando el enlace se libera, vuelve al estado Espera_CNX.

	Estados		
Señales	a	b	c
-	b		
DL_ESTABLISH_IND		c	
DL_DATA_IND			c
DL_RELEASE_IND			b
BROADCAST		b	c

(a)



(b)

Figura J.30: (a) Transiciones posibles y (b) Diagrama de estados para el proceso EMU_NWK_FT.

J.4.3 Emulador EMU_NWK_PT

El emulador del Nivel de Red de la Terminación Portátil incorpora la funcionalidad necesaria para poder ejecutar los Casos de Prueba del Juego de Pruebas SP_DLC_PT_Gap.mp sobre el Nivel de Enlace de dicha Terminación. Se comunica con el Nivel de Enlace a través de las dos interfaces que ofrece (SAPB_PT y SAP0_PT) (Figura J.31) y con el bloque de gestión a través de la interfaz LLME_NWK. El operador puede controlar la operación de la Terminación Portátil a través de la ruta ENV_EMU_NWK. Las acciones que el operador puede solicitar son crear conexión (CREAR_CONEX), enviar datos (ENVIAR_DATOS) y liberar conexión (CERRAR_CONEX).

Durante la inicialización, se envía la identidad temporal, TPUI, al bloque de gestión LLME_PT. Se crea una conexión cuando lo solicita el operador (CREAR_CONEX) tras una petición de las pruebas, o cuando se recibe la señal de difusión (DL_BROADCAST_IND). Mientras el enlace se encuentra establecido se pueden enviar y recibir datos.

El emulador, al igual que para la Terminación Fija, contiene un único proceso. Su diagrama de estados y las transiciones posibles en cada uno de ellos para las señales válidas se muestran en la Figura J.32. El comportamiento de cada estado es el siguiente:

- Escucha: Es el estado inicial, a la espera de recibir la información de difusión o la orden del operador de crear una conexión.

SPICUM servicio de publicaciones

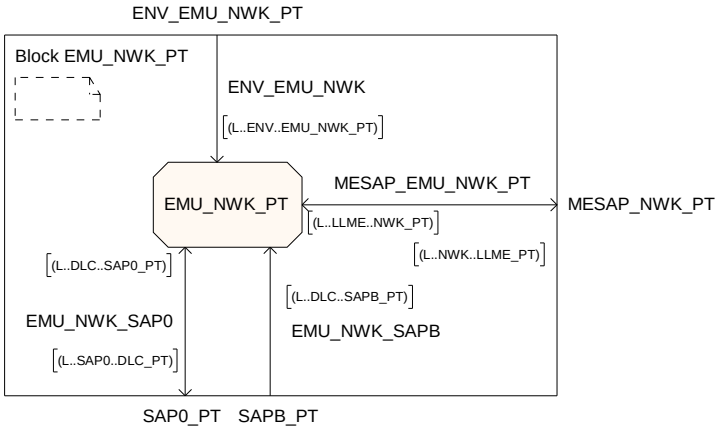
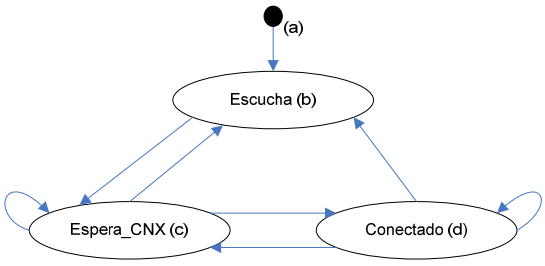


Figura J.31: Modelo del emulador del Nivel de Red de la Terminación Portátil.

- `Espera_CNX`: Cuando se ha recibido la confirmación del establecimiento del enlace, pasa al estado `Conectado`; si no, vuelve a `Escucha`.
- `Conectado`: Permite recibir y enviar datos. Al liberar la conexión vuelve al estado `Escucha`.

	Estados			
Señales	a	b	c	d
-	b			
DL_BROADCAST_IND		c		
DL_ESTABLISH_CFM			d	
DL_DATA_IND				d
DL_RELEASE_IND			b	b
CREAR_CONEX		c	c	c
ENVIAR_DATOS				d
CERRAR_CONEX				b

(a)



(b)

Figura J.32: (a) Transiciones posibles y (b) Diagrama de estados para el proceso `EMU_NWK_PT`.

J.4.4 Emulador `EMU_DLC_FT`

Este bloque emula el comportamiento del Nivel de Enlace de la Terminación Fija. Contiene dos procesos (Figura J.33), uno para el envío de mensajes de difusión (`EMU_DLC_BS_FT`), y otro para gestionar las conexiones y el envío de datos (`EMU_DLC_DATOS_FT`) [CESP99].

El proceso `EMU_DLC_BS_FT` envía periódicamente (`T1`) los mensajes de difusión. Como para las pruebas del Nivel MAC la información del canal B_S carece de importancia, se rellena con los valores por defecto. Desde el simulador se inicia el envío de estos mensajes, y también se puede ordenar su envío sin esperar a que venza el temporizador. El diagrama de estados de este proceso y las transiciones posibles en cada uno de ellos para las señales válidas se muestran en la Figura J.34. El comportamiento de cada estado es el siguiente:

- `Wait`: Recibe las señales del simulador y envía periódicamente los mensajes de difusión.

SPICUM servicio de publicaciones

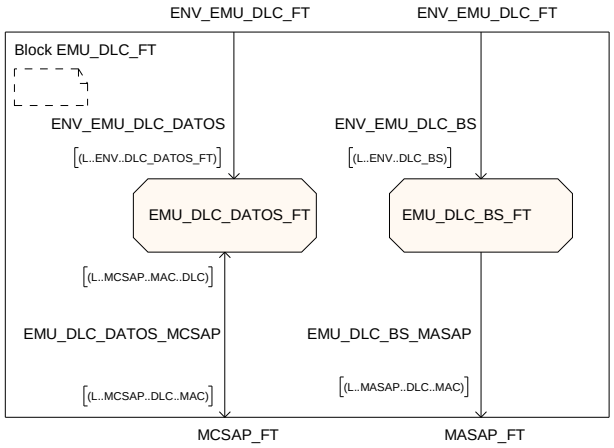


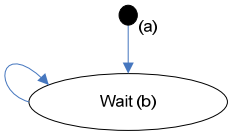
Figura J.33: Modelo del emulador del Nivel de Enlace de la Terminación Fija.

El proceso `EMU_DLC_DATOS_FT` envía datos cuando se solicita desde el simulador (`ENVIAR_DATOS`) o cuando el Nivel MAC le indica que está preparado para enviar más datos (`MAC_CO_DTR_IND`). Se lleva una cuenta del número de envíos y recepciones realizados. El diagrama de estados de este proceso y las transiciones posibles en cada uno de ellos para las señales válidas se muestran en la Figura J.35. El comportamiento de cada estado es el siguiente:

- `Wait`: Espera la solicitud de conexión.
- `Espera_Comando`: Recibe solicitudes de envío de datos del Nivel DLC, datos de la entidad remota y solicitudes de traspaso de la conexión.
- `Connection_Ho`: Realiza el traspaso de la conexión.

	Estados	
Señales	a	b
-	b	
T1		b
ENVIAR_UN_PAGING		b
ENVIAR_PAGING		b

(a)



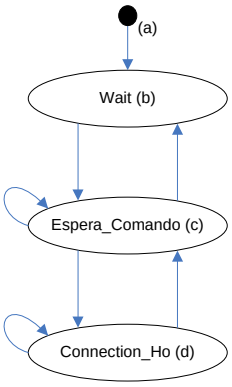
(b)

Figura J.34: (a) Transiciones posibles y (b) Diagrama de estados para el proceso `EMU_DLC_BS_FT`.

SPICUM servicio de publicaciones

Señales	Estados			
	a	b	c	d
-	b			
MAC_CON_IND		c	d	
MAC_DIS_IND			b	c
MAC_CO_DTR_IND			c	d
MAC_CO_DATA_IND			c	d
ENVIAR_DATOS			c	d

(a)



(b)

Figura J.35: (a) Transiciones posibles y (b) Diagrama de estados para el proceso `EMU_DLC_DATOS_FT`.

J.4.5 Emulador `EMU_DLC_PT`

Este bloque emula el comportamiento del Nivel de Enlace de la Terminación Portátil. Al igual que para la Terminación Fija, contiene dos procesos (Figura J.36), uno para recibir los mensajes de difusión (`EMU_DLC_BS_PT`), y otro para gestionar las conexiones y recibir y enviar datos (`EMU_DLC_DATOS_PT`). Este emulador funciona de forma muy parecida al de la Terminación Fija; las principales diferencias son que a través del simulador se puede solicitar la creación de conexiones de tráfico y su traspaso, mientras que la Terminación Fija simplemente reaccionaba a estas solicitudes recibidas de la entidad remota.

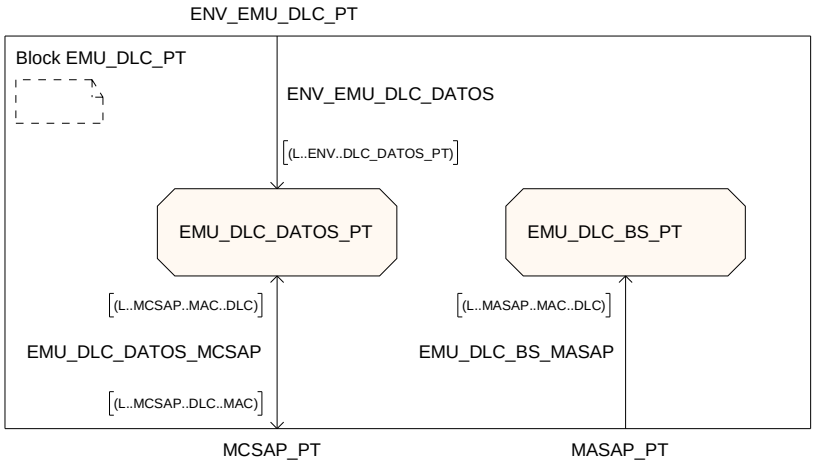


Figura J.36: Modelo del emulador del Nivel DLC de la Terminación Portátil.

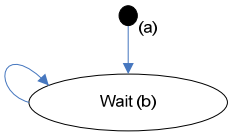
El proceso `EMU_DLC_BS_PT` se limita a recibir los mensajes de difusión. El diagrama de estados de este proceso y las transiciones posibles en cada uno de ellos para las señales válidas se muestran en la Figura J.37. El comportamiento de cada estado es el siguiente:

- `Wait`: Recibe los mensajes de difusión.

SPICUM servicio de publicaciones

	Estados	
Señales	a	b
-	b	
MAC_PAGE_IND		b

(a)



(b)

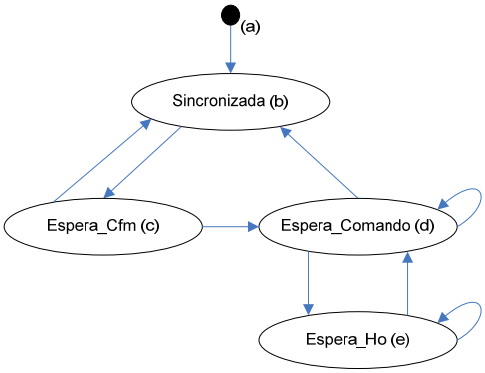
Figura J.37: (a) Transiciones posibles y (b) Diagrama de estados para el proceso EMU_DLC_BS_PT.

El proceso EMU_DLC_DATOS_PT gestiona (creación, traspaso y liberación) las conexiones de tráfico y permite la transferencia de datos. La creación (CREAR_CONEX) de una conexión se solicita desde el simulador; desde el mismo también se puede solicitar el envío de datos (ENVIAR_DATOS) y la liberación de la conexión (CERRAR_CANAL_DATOS). El diagrama de estados de este proceso y las transiciones posibles en cada uno de ellos para las señales válidas se muestran en la Figura J.38. El comportamiento de cada estado es el siguiente:

- Sincronizada: Espera la orden para crear un canal de tráfico.
- Espera_Cfm: Espera la confirmación de la creación del canal de tráfico, o su rechazo.
- Espera_Comando: Recibe solicitudes de envío de datos del Nivel DLC, datos de la entidad remota y solicitudes, desde el simulador, de traspaso de la conexión.
- Espera_Ho: Espera la respuesta a la solicitud de traspaso de la conexión.

	Estados				
Señales	a	b	c	d	e
-	b				
MAC_CON_CFM			d		e
MAC_DIS_IND			b	b	d
MAC_CO_DTR_IND				d	e
MAC_CO_DATA_IND				d	e
ENVIAR_DATOS				d	e
CREAR_CANAL_DATOS		c			
CERRAR_CANAL_DATOS				b	
CONN_HO				e	
FIN_CONN_HO					d

(a)



(b)

Figura J.38: (a) Transiciones posibles y (b) Diagrama de estados para el proceso EMU_DLC_DATOS_PT.

J.4.6 Emulador EMU_MAC

El bloque emulador del Nivel MAC [DOMI99] contiene dos procesos (Figura J.39), cada uno de los cuales emula la funcionalidad mínima correspondiente al Nivel MAC de cada Terminación, Fija (EMU_MAC_FT) o Portátil (EMU_MAC_PT). En esencia, se trata de un 'tubo' que traduce las primitivas entre ambos extremos. Cada proceso ha sido modelado, en realidad, como un tipo de proceso, para poder incluirlo en el paquete Sub_DLC.

SPICUM servicio de publicaciones

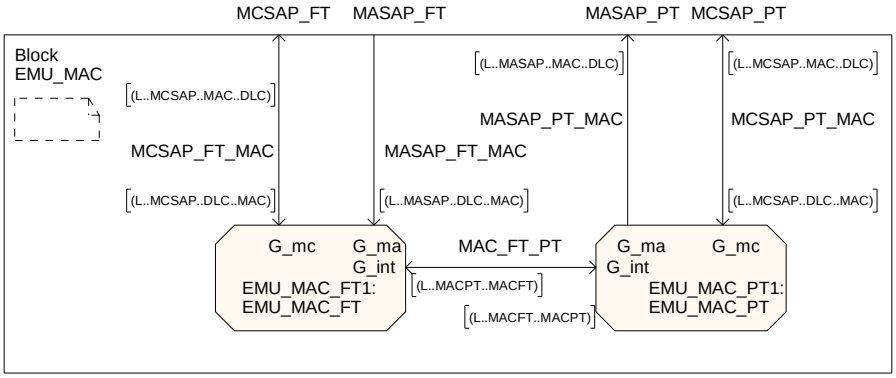


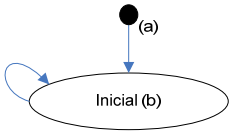
Figura J.39: Modelo del emulador del Nivel de Acceso al Medio.

Cada proceso traduce las peticiones que recibe del Nivel DLC local a primitivas de indicación (creación de conexión[10], transferencia de datos, liberación de conexión y *paging*[11]). El proceso receptor sustituye el campo de identificación `mcei` por el utilizado localmente y reenvía la primitiva recibida al Nivel DLC. Ambos procesos incluyen un único estado (`Inicial`), donde se procesan todas las primitivas.

El diagrama de estados del proceso `EMU_MAC_FT` y las transiciones posibles en cada uno de ellos para las señales válidas se muestran en la Figura J.40. El diagrama de estados del proceso `EMU_MAC_PT` y las transiciones posibles en cada uno de ellos para las señales válidas se muestran en la Figura J.41.

	Estados	
Señales	a	b
-	b	
MAC_CON_IND		b
MAC_CO_DATA_REQ		b
MAC_CO_DATA_IND		b
MAC_DIS_REQ		b
MAC_DIS_IND		b
MAC_PAGE_REQ		b

(a)

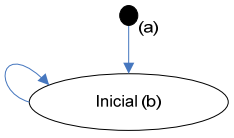


(b)

Figura J.40: (a) Transiciones posibles y (b) Diagrama de estados para el proceso `EMU_MAC_FT`.

	Estados	
Señales	a	b
-	b	
MAC_CON_REQ		b
MAC_CO_DATA_REQ		b
MAC_CO_DATA_IND		b
MAC_DIS_REQ		b
MAC_DIS_IND		b
MAC_PAGE_IND		b

(a)



(b)

Figura J.41: (a) Transiciones posibles y (b) Diagrama de estados para el proceso `EMU_MAC_PT`.

[10] Sólo la Terminación Portátil.

[11] Sólo la Terminación Fija.

SPICUM servicio de publicaciones

J.4.7 Emulador EMU_PHY

Este bloque emula el comportamiento del Nivel Físico conjuntamente para ambas Terminaciones.. Incorpora la funcionalidad mínima para comunicar una Terminación Fija con una Terminación Portátil; básicamente, convierte las peticiones de un extremo en las correspondientes indicaciones a recibir en el otro.

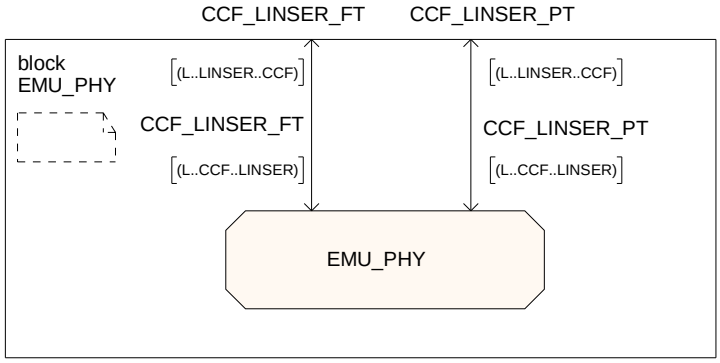
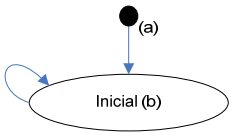


Figura J.42: Modelo del emulador del Nivel de Físico.

El emulador está formado (Figura J.42) por un único proceso con un único estado, en el cual se procesan todas las señales. El diagrama de estados y las transiciones posibles para las señales válidas se muestran en la Figura J.43.

	Estados	
Señales	a	b
-	b	
CSF_INIT		b
CSF_SCAN_REQ		b
CSF_DBC_REQ		b
CSF_CO_DATA_REQ		b
CSF_TBC_RES_REQ		b
CSF_PAGE_INFO_SHORT_REQ		b
CSF_DIS_REQ		b

(a)



(b)

Figura J.43: (a) Transiciones posibles y (b) Diagrama de estados para el proceso EMU_PHY.

SPICUM servicio de publicaciones

Apéndice

K

APÉNDICE K: MODELADO DEL NIVEL DE RED DE DECT

En este Apéndice se describe el modelo del Nivel de Red de sistemas DECT [ETS 300 175-5] que se ha realizado siguiendo la Metodología de Diseño M-SOMT descrita en el Capítulo 7. La funcionalidad se ha agrupado en cuatro procesos[1]: Control de la Llamada (CC – *Call Control*), Gestión de la Movilidad (MM – *Mobility Management*), Entidad de Control del Enlace (LCE – *Link Control Entity*) y Entidad de Gestión de los Niveles Inferiores (LLME – *Lower Layer Management Entity*). Cada uno de estos procesos se ha modelado mediante un tipo de proceso común a ambas Terminaciones (Fija y Portátil) que es heredado por el tipo de proceso correspondiente según sea una Terminación Fija (prefijo `F_`) o una Terminación Portátil (prefijo `P_`). Por ejemplo, el Control de la Llamada se modela en los tipos de proceso `CC`, `F_CC` y `P_CC`.

Estos tipos de proceso se instancian para construir el Nivel de Red, modelado como un tipo de bloque, para la Terminación Fija (`F_NWK` formado por `F_CC`, `F_MM` y `F_LCE`) y para la Terminación Portátil (`P_NWK` formado por `P_CC`, `P_MM` y `P_LCE`). La funcionalidad de gestión (`F_LLME`, `P_LLME`) se ha incluido en un bloque propio.

El modelado se ha estructurado en tres paquetes:

- `Comun`: Contiene tipos de proceso base y procedimientos comunes para ambas Terminaciones.
- `ParteFija`: Contiene los tipos de proceso de la Terminación Fija.
- `PartePortatil`: Contiene los tipos de proceso de la Terminación Portátil.

A continuación se describe cada uno de los tipos de bloque y de proceso.

K.1 Estructura del Nivel de Red

La estructura del Nivel de Red se muestra en la Figura K.1 para ambas Terminaciones. Esta estructura sigue las directrices sugeridas en la especificación del Nivel de Red [ETS

[1] No se incluye una descripción del proceso `Card` pues su función es únicamente servir de almacenamiento de parámetros del sistema. Esta funcionalidad se ha integrado finalmente dentro de la Entidad de Gestión de los Niveles Inferiores (`LLME`).

SPICUM servicio de publicaciones

300 175-5]. La Tabla K.1 y la Tabla K.2 describen el propósito de cada una de las interfaces.

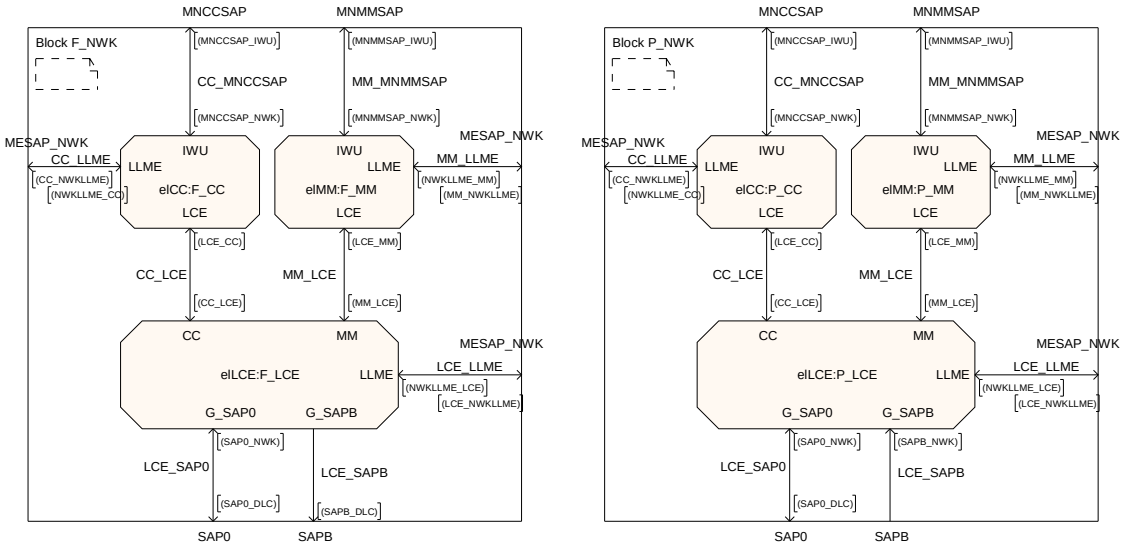


Figura K.1: Estructura del Nivel de Red para la (a) Terminación Fija y la (b) Terminación Portátil.

Tabla K.1: Interfaces externas del Nivel de Red.

SAP	Entidad Remota	Descripción
MNCCSAP	IWU	Interfaz de servicios de Control de la Llamada.
MMSAP	IWU	Interfaz de servicios de Gestión de la Movilidad.
SAP0	DLC	Proporciona servicios orientados a conexión.
SAPB	DLC	Interfaz de los servicios de difusión.
MESAP_NWK	LLME	Acceso a los servicios de la entidad de gestión para el Nivel de Red.

Tabla K.2: Interfaces internas del Nivel de Red.

Canal	Entidades	Descripción
CC_LCE	CC – LCE	Comunicación entre Control de la Llamada y la Entidad de Control del Enlace.
MM_LCE	MM – LCE	Comunicación entre Gestión de la Movilidad y la Entidad de Control del Enlace.

SPICUM servicio de publicaciones

K.2 Entidades del Nivel de Red

K.2.1 Control de la Llamada

El Control de la Llamada es el conjunto de procedimientos que gobiernan el establecimiento, mantenimiento y liberación de circuitos conmutados, encargándose de la señalización necesaria para realizar estas acciones. Cada llamada es asociada a un determinado servicio del plano de usuario por la Entidad de Gestión (`LLME`).

El comportamiento de este proceso es diferente según sea el iniciador o el destinatario de la llamada. Si el rol es el de iniciador (envío de `CC_SETUP`), se pasa al estado `CALL_PRESENT`, donde se espera la respuesta a la petición de establecimiento. Cuando se actúa como destinatario (recepción de `CC_SETUP`), se envía la indicación al nivel superior para que acepte o rechace la llamada. El establecimiento de la llamada puede iniciarse simultáneamente en ambos extremos, produciéndose una colisión; este caso también se ha contemplado en el diseño. Una vez se ha completado el establecimiento se pasa al estado `ACTIVO`. Tras la liberación se vuelve al estado de reposo (`REPOSO`).

K.2.1.1 Tipo Base

El tipo de proceso base (`CC`) incluido en el paquete `Comun` define las puertas de comunicación y un conjunto de procedimientos comunes a ambas Terminaciones (Tabla K.3).

Tabla K.3: Procedimientos definidas en el tipo de proceso `CC`.

`PeticionDeServicio_CC`	`LiberacionCompletaDestino`
`ConfirmarPlanoU`	`LiberacionNormalOrigen_CC`
`DesconectarPlanoU`	`LiberacionCompletaDestino_CC`
`PasoInformacionDestino_CC`	`LiberacionAnormalDestino_CC`
`PasoInformacionOrigen_CC`	`Vencimiento`
`RechazoLlamada_IWU`	`Formar<PDU>`

K.2.1.2 Terminación Fija

El diagrama de estados de este proceso (`F_CC`) y las transiciones posibles en cada uno de ellos para las señales válidas se muestran en la Figura K.2. El comportamiento de cada estado es el siguiente:

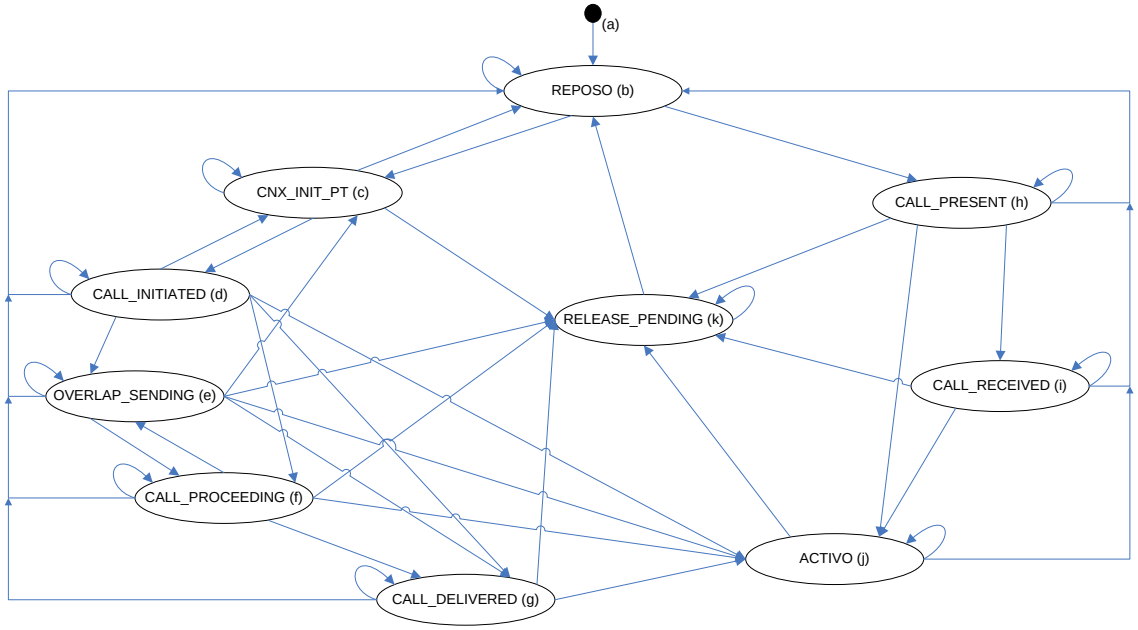
- `REPOSO`: Es el estado en el cual no hay una conexión en procesamiento.
- Estados correspondientes a la fase de establecimiento de una conexión iniciada por la Terminación Portátil:
  - `CNX_INIT_PT`: Espera la señal de solicitud de establecimiento de conexión tras una solicitud de la Entidad de Control del Enlace.
  - `CALL_INITIATED`: Estado en el cual se espera la aceptación de la conexión por parte del nivel superior.

SPICUM servicio de publicaciones

- OVERLAP_SENDING: Estado en el cual se espera más información del otro extremo.
- CALL_PROCEEDING: Se ha indicado que se ha recibido toda la información necesaria para el establecimiento pero todavía no se ha aceptado.
- CALL_DELIVERED: Estado en el cual se ha activado el aviso de llamada entrante.

	Estados										
Señales	a	b	c	d	e	f	g	h	i	j	k
-	b										
LCE_ESTABLISH_IND		c									
CC_STATUS_REQ		b	c	d	e	f	g	h	i	j	k
CC_SETUP			d								
MNCC_REJECT_REQ				c	c						
MNCC_SETUP_ACK_REQ				e							
MNCC_CALL_PROC_REQ				f	f						
MNCC_CONNECT_REQ				j	j	j	j				
MNCC_ALERT_REQ				g	g	g					
LLME_RESTART_REQ				d							
CC_INFO					e					j	
F_CC.01					k						
CC_INFO						e					b
MNCC_SETUP_REQ		h									
CC_ALERTING								i			
CC_CONNECT								j	j		
F_CC.03								b			
MNCC_INFO_REQ									i	j	
CC.02											b
CC_RELEASE_CFM			b	b	b	b	b	b	b	b	b
CC_RELEASE			b	b	b	b	b	b	b	b	b
MNCC_RELEASE_REQ			k			k	k	k	k	k	b
LCE_RELEASE_IND			b	b	b	b	b	b	b	b	b

(a)



(b)

Figura K.2: (a) Transiciones posibles y (b) Diagrama de estados para el proceso F_CC.

SPICUM servicio de publicaciones

- Estados correspondientes a la fase de establecimiento de una conexión iniciada por la Terminación Fija:
    - `CALL_PRESENT`: Se ha enviado el mensaje de establecimiento pero aún no se ha recibido contestación.
    - `CALL_RECEIVED`: Estado en el cual se ha activado el aviso de llamada entrante en el destinatario remoto, pero el usuario aún no ha contestado.
- `ACTIVO`: Es el estado en el que la llamada se encuentra establecida.
- `RELEASE_PENDING`: Estado intermedio durante la fase de liberación, en el que se ha enviado la petición de liberación pero aún no se ha recibido respuesta.

Además, en este proceso se definen procedimientos específicos para la Terminación Fija.

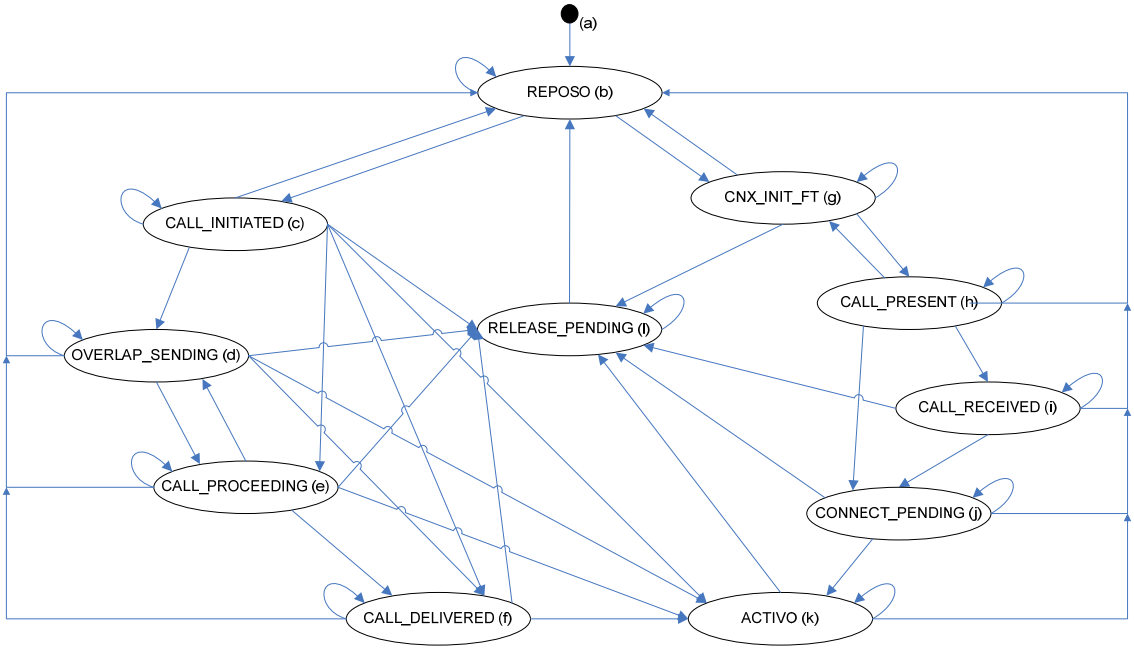
K.2.1.3 Terminación Portátil

El diagrama de estados de este proceso (`P_CC`) y las transiciones posibles en cada uno de ellos para las señales válidas se muestran en la Figura K.3. El comportamiento de cada estado es equivalente al proceso `F_CC`, aunque vistos desde el otro extremo. En este caso, desaparece el estado `CNX_INIT_PT`, siendo sustituido por el estado `CNX_INIT_FT`. También se definen procedimientos específicos para la Terminación Portátil.

	Estados											
Señales	a	b	c	d	e	f	g	h	i	j	k	l
-	b											
CC_STATUS_REQ		b	c	d	e	f	g	h	i	j	k	l
MNCC_SETUP_REQ		c										
CC_SETUP_ACK			d									
P_CC.03			b									
CC_CALL_PROC			e	e								
CC_ALERTING			f	f	f							
CC_NOTIFY			c									
CC_CONNECT			k	k	k	k						
MNCC_INFO_REQ				d	e						k	
LCE_ESTABLISH_IND		g										
CC_SETUP							h					
MNCC_REJECT_REQ								g				
MNCC_ALERT_REQ								i				
MNCC_CONNECT_REQ								j	j			b
CC_INFO									i		k	
CC_CONNECT_ACK										k		
P_CC.05										l		
CC.02												b
CC_RELEASE_CFM			b	b	b	b	b	b	b	b	b	b
CC_RELEASE			b	b	b	b	b	b	b	b	b	b
MNCC_RELEASE_REQ			l	l	l	l	l		l	l	l	
LCE_RELEASE_IND			b	b	b	b	b	b	b	b	b	b

(a)

SPICUM servicio de publicaciones



(b)

Figura K.3: (a) Transiciones posibles y (b) Diagrama de estados para el proceso `P_CC`.

K.2.2 Gestión de la Movilidad

La Gestión de la Movilidad es la entidad encargada de la identificación, autentificación y localización de terminales y, en general, de las tareas necesarias para permitir la provisión segura de los servicios DECT y su protección frente a ataques fraudulentos. Los procedimientos llevados a cabo por esta entidad se pueden agrupar en siete categorías: identificación, autentificación, localización, asignación de clave, cifrado, recogida de información y derechos de acceso.

Este proceso ha sido diseñado como un conjunto de transacciones alternativas que empiezan en el estado `REPOSO`, al cual retornan una vez finalizadas. Mientras se está atendiendo a una solicitud, cualquier otra petición permanece en espera.

K.2.2.1 Tipo Base

El tipo de proceso base (`MM`) incluido en el paquete `Comun` define las puertas de comunicación y un conjunto de procedimientos comunes a ambas Terminaciones (Tabla K.4).

Tabla K.4: Procedimientos definidas en el tipo de proceso `MM`.

`PeticionDeServicio_MM`	`Formar<PDU>`
`CalcularRES`	

SPICUM servicio de publicaciones

K.2.2.2 Terminación Fija

El diagrama de estados de este proceso (`F_MM`) y las transiciones posibles en cada uno de ellos para las señales válidas se muestran en la Figura K.4. El comportamiento de cada estado es el siguiente:

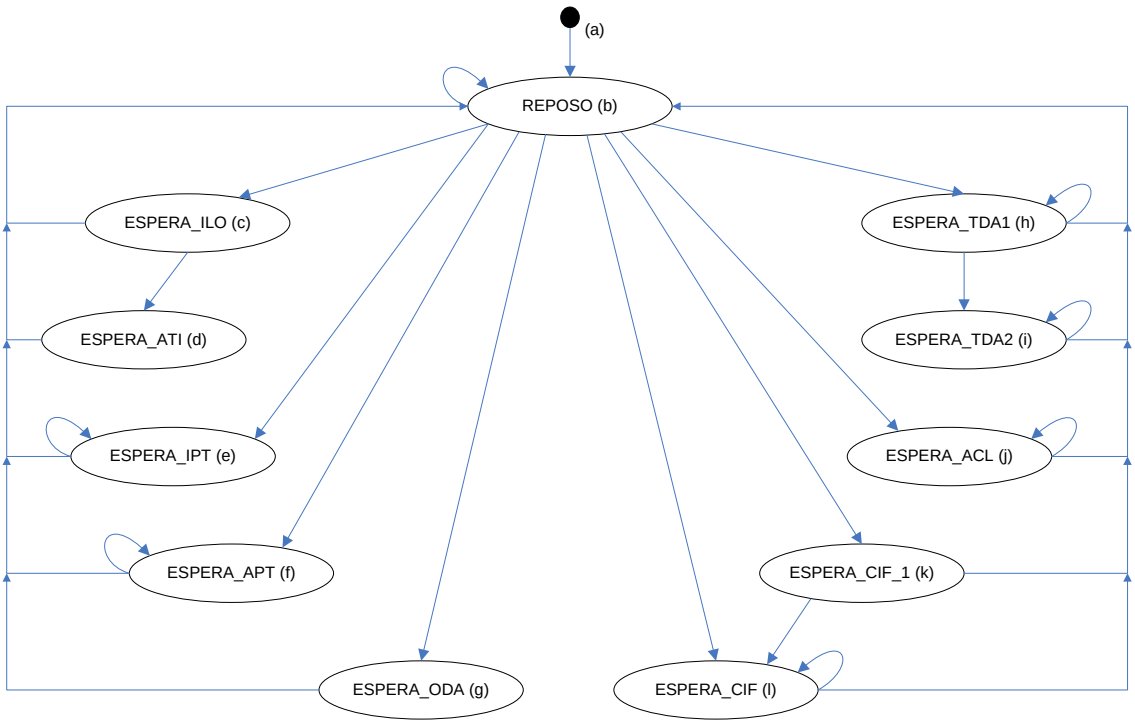
- `REPOSO`: Es el estado en el que no se está procesando ninguna transacción.
- `ESPERA_ILO`: Estado intermedio para el registro de localización.
- `ESPERA_ATI`: Estado intermedio para el registro de localización.
- `ESPERA_IPT`: Estado de espera de parámetros de identificación de la Terminación Portátil.
- `ESPERA_APT`: Estado intermedio durante la autentificación de la Terminación Portátil.
- `ESPERA_ODA`: Estado perteneciente a la transacción de obtención de derechos de acceso.
- `ESPERA_TDA1`: Estado intermedio durante el procedimiento de terminación de derechos de acceso cuando es iniciada por la Terminación Fija.
- `ESPERA_TDA2`: Estado intermedio durante el procedimiento de terminación de derechos de acceso cuando es iniciada por la Terminación Fija.
- `ESPERA_ACL`: Estado intermedio de la asignación de la clave de cifrado.
- `ESPERA_CIF`: Estado correspondiente a la transacción inicio de cifrado cuando es iniciada por la Terminación Fija.
- `ESPERA_CIF_1`: Estado correspondiente a la transacción inicio de cifrado cuando es iniciada por la Terminación Portátil.

Además, en este proceso se definen procedimientos específicos para la Terminación Fija.

SPICUM servicio de publicaciones

Señales	Estados											
	a	b	c	d	e	f	g	h	i	j	k	l
-	b											
LOCATE_REQUEST		c										
MM_LOCATE_RES			b,d									
TEMPORARY_ID_ASSIGN_ACK				b								
TEMPORARY_ID_ASSIGN_REJECT				b								
MM_ident.1				b								
MM_IDENTITY_REQ		e										
IDENTITY_REPLY					b							
MM_ident.2					b,e							
MM_AUTHENTICATE_REQ		f										
AUTH_REPLY						b						
AUTH_REJECT						b				b		
MM_auth.1						b,f						
MM_auth.2						b,f						
AUTH_REQUEST		b				f		i		b		
MM_INFO_REQ		b										
ACCESS_RIGHTS_REQUEST		g										
MM_ACCESS_RIGHTS_RES							b					
MM_ACCESS_RIGHTS_TERMINATE_REQ		h										
MM_access.2								b,h	b,i			
ACCESS_RIGHTS_TERM_ACCEPT									b			
ACCESS_RIGHTS_TERM_REJECT									b			
MM_KEY_ALLOCATE_REQ		j										
MM_key.1										b,j		
CIPHER_SUGGEST		k										
MM_CIPHER_RES											b,l	
CIPHER_REJECT												b
DL_ENCRYPT_IND												b
MM_CIPHER_REQ		l										
MM_cipher.1												b,l
LCE_RELEASE_IND		b	b	b	b	b	b	b	b	b	b	b

(a)



(b)

Figura K.4: (a) Transiciones posibles y (b) Diagrama de estados para el proceso F_MM.

SPICUM servicio de publicaciones

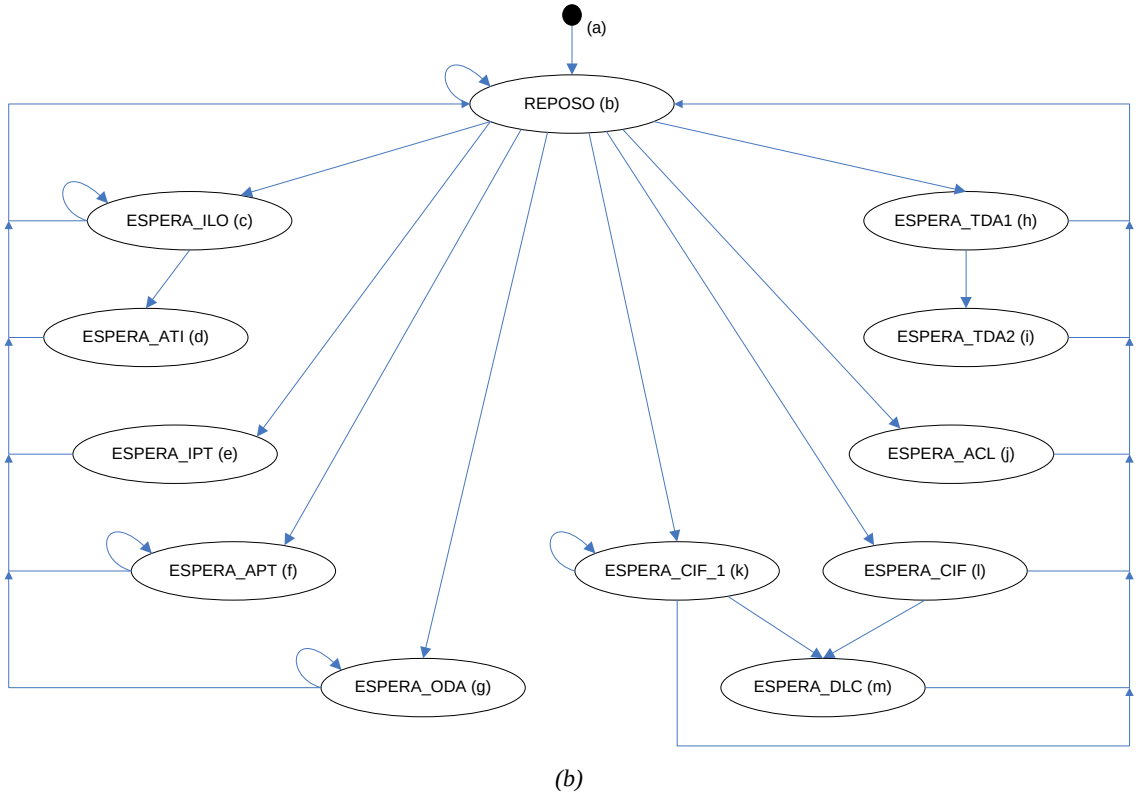
K.2.2.3 Terminación Portátil

El diagrama de estados de este proceso (P_CC) y las transiciones posibles en cada uno de ellos para las señales válidas se muestran en la Figura K.5. El comportamiento de cada estado es equivalente a los del proceso F_MM. Aparece un nuevo estado, ESPERA_DLC, donde se espera la confirmación para el cambio de cifrado por parte del Nivel de Enlace. También se definen procedimientos específicos para la Terminación Portátil.

	Estados												
Señales	a	b	c	d	e	f	g	h	i	j	k	l	m
-	b												
MM_LOCATE_REQ		c											
LLME_LOCATION_UPDATE		c											
LOCATE_ACCEPT			d										
LOCATE_REJECT			b										
MM_locate.1			b,c										
MM_IDENTITY_ASSIGN_RES				b									
IDENTITY_REQUEST		e											
MM_IDENTITY_RES					b								
MM_AUTHENTICATE_REQ		b				f		i					
AUTH_REQUEST		f											
MM_AUTHENTICATE_RES						b							
MM_INFO_SUGGEST		b											
MM_ACCESS_RIGHTS_REQ		g											
ACCESS_RIGHTS_ACCEPT							b						
ACCESS_RIGHTS_REJECT							b						
MM_access.1							b,g						
ACCESS_RIGHTS_TERM_REQUEST		h											
MM_ACCESS_RIGHTS_TERMINATE_RES									b				
KEY_ALLOCATE		b,j											
AUTH_REPLY										b			
AUTH_REJECT										b			
MM_auth.1										b			
MM_CIPHER_REQ		k											
MM_cipher.2											b,k		
CIPHER_REQUEST		l									m		
MM_CIPHER_RES												b,m	
DL_ENCRYPT_CFM													b
CIPHER_REJECT											b		
LCE_RELEASE_IND		b	b	b	b	b	b	b	b	b	b	b	b

(a)

SPICUM servicio de publicaciones



(b)

Figura K.5: (a) Transiciones posibles y (b) Diagrama de estados para el proceso `P_MM`.

K.2.3 Entidad de Control del Enlace

La Entidad de Control del Enlace (LCE) es la entidad inferior del Nivel de Red. Ofrece un servicio de datos a las entidades superiores con objeto de que puedan comunicarse con sus pares en el sistema remoto. Esta entidad se encarga de establecer, mantener y liberar los enlaces del plano de control y de enrutar los mensajes que transportan entre las distintas entidades.

El comportamiento de esta entidad es tal que si el establecimiento del enlace es solicitado por el nivel superior de la Terminación Fija, se transmite un mensaje de búsqueda dentro de una primitiva de difusión (`DL_BROADCAST_REQ`); si se inicia debido a una solicitud del nivel superior de la Terminación Portátil se envía un mensaje `DL_ESTABLISH_REQ`. En ambos casos se pasa al estado `ESTABLISH_PENDING`. Una vez confirmado el establecimiento se pasa al estado `LINK_ESTABLISHED`, donde se puede realizar el intercambio de información entre las entidades remotas; los datos provenientes de `CC` o `MM` se encapsulan en primitivas `DL_DATA_REQ`. Si se produce una pérdida anormal del enlace, se intenta su restablecimiento, esperando la confirmación o no en el estado `REESTABLISH_PENDING`.

K.2.3.1 Tipo Base

El tipo de proceso base (`LCE`) incluido en el paquete `Comun` define las puertas de comunicación y un conjunto de procedimientos comunes a ambas Terminaciones (Tabla K.5).

SPICUM servicio de publicaciones

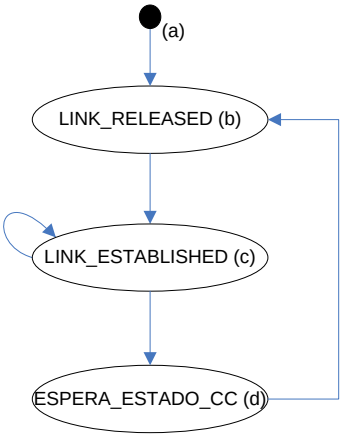
Tabla K.5: Procedimientos definidas en el tipo de proceso `LCE`.

`Link_Establish`	`LiberacionNormal_LCE`
`DesmontarMESSAGE_UNIT`	`LiberacionAnormal_LCE`
`Formar<PDU>`	

En esta entidad se ha definido también la respuesta a diversas señales en el estado `LINK_ESTABLISHED`, que es en el que se produce todo el intercambio de datos extremo a extremo. Las transiciones posibles en este estado para las señales válidas se muestran en la Figura K.6. El diagrama de estados muestra otros dos estados adicionales cuyo tratamiento se realiza realmente en los tipos de proceso especializados; estos estados se describen en el siguiente apartado.

	Estados			
Señales	a	b	c	d
-	b			
LCE_RELEASE_REQ			d	
LCE_ESTABLISH_REQ		c	c	
DL_RELEASE_IND			d	
CC_RELEASE			c	
CC_RELEASE_CFM			c	
CC_SETUP			c	
DL_RELEASE_CFM				b
MM_OUT_OF_SCOPE			c	
AUTH_REJECT			c	
AUTH_REPLY			c	
AUTH_REQUEST			c	
CC_ALERTING			c	
CC_CONNECT			c	
CC_INFO			c	
CC_OUT_OF_SCOPE			c	

(a)



(b)

Figura K.6: (a) Transiciones posibles y (b) Diagrama de estados para el proceso `LCE`.

K.2.3.2 Terminación Fija

La Entidad de Control del Enlace para la Terminación Fija redefine el comportamiento modelado en el paquete `Comun` ampliando el número de señales a las que responde. También añade la vía de comunicación correspondiente al servicio de difusión (`G_SAPB`)[2]. El diagrama de estados de este proceso (`F_LCE`) y las transiciones posibles en cada uno de ellos para las señales válidas se muestran en la Figura K.7; no se han incluido las transiciones definidas en el tipo base. El comportamiento de cada estado, salvo el estado `LINK_ESTABLISHED`, es el siguiente:

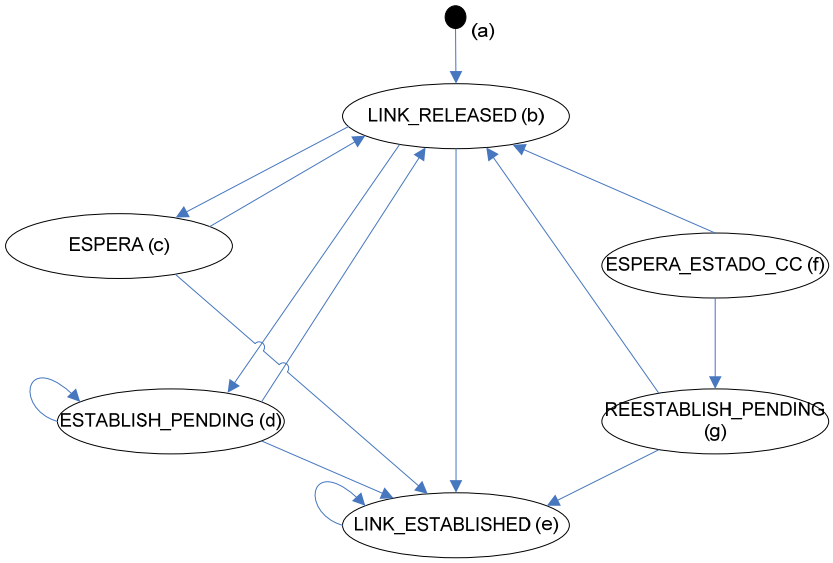
- `LINK_RELEASED`: Estado inicial en el que se esperan solicitudes de establecimiento de nuevos enlaces.
- `ESPERA`: Estado en el que se espera la confirmación de establecimiento de un nuevo enlace por parte del Nivel de Enlace.

[2] Esta vía de comunicación no se ha incluido en el tipo base porque tiene direccionalidad diferente según sea la Terminación Fija (saliente) o la Terminación Portátil (entrante).

SPICUM servicio de publicaciones

Señales	Estados						
	a	b	c	d	e	f	g
-	b						
DL_ESTABLISH_IND		e,c		b,e			e
DL_DATA_IND			e		e		
LCE.05			b				
CC_STATUS_CFM						b,g	
LCE.04							b
LCE_ESTABLISH_REQ		d					
LCE.03				b,d			
ACCESS_RIGHTS_ACCEPT					e		
ACCESS_RIGHTS_REJECT					e		
ACCESS_RIGHTS_TERM_REQUEST					e		
CC_CALL_PROC					e		
CC_CONNECT_ACK					e		
CC_NOTIFY					e		
CC_SETUP_ACK					e		
CIPHER_REQUEST					e		
CIPHER_REJECT					e		
IDENTITY_REQUEST					e		
KEY_ALLOCATE					e		
LOCATE_ACCEPT					e		
LOCATE_REJECT					e		
MM_INFO_SUGGEST					e		
DL_ENCRYPT_IND					e		
DL_ENC_KEY_REQ					e		

(a)



(b)

Figura K.7: (a) Transiciones posibles y (b) Diagrama de estados para el proceso F_LCE.

- ESPERA_ESTADO_CC: Punto en el que se espera información sobre el estado de la entidad de Control de la Llamada.
- REESTABLISH_PENDING: Se notifica la pérdida del enlace pero si se recibe una confirmación adecuada del Nivel de Enlace se retorna al estado LINK_ESTABLISHED.
- ESTABLISH_PENDING: Se está a la espera de recibir la respuesta, confirmación o rechazo del Nivel de Enlace sobre la petición de establecimiento de un nuevo enlace.

SPICUM servicio de publicaciones

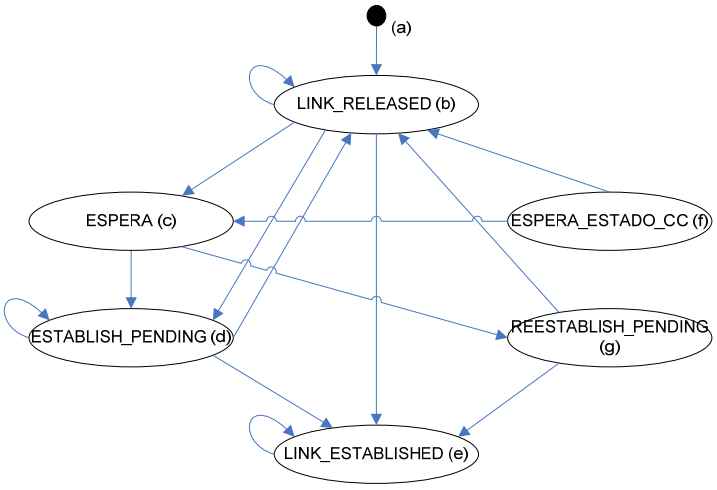
Además, en este proceso se definen procedimientos específicos para la Terminación Fija.

K.2.3.3 Terminación Portátil

La Entidad de Control del Enlace para la Terminación Portátil también redefine el comportamiento modelado en el paquete `Comun` al estilo de la Terminación Fija. Igualmente, añade la vía de comunicación correspondiente al servicio de difusión (`G_SAPB`)[3], aunque con sentido opuesto. El diagrama de estados de este proceso (`P_LCE`) y las transiciones posibles en cada uno de ellos para las señales válidas se muestran en la Figura K.8; tampoco se han incluido las transiciones definidas en el tipo base. El comportamiento de los estados es equivalente al de la Terminación Fija. También se definen procedimientos específicos para la Terminación Portátil.

	Estados						
Señales	a	b	c	d	e	f	g
-	b						
LCE_ESTABLISH_REQ		c					
LLME_IDENTIFIER_RES			d,g				
DL_BROADCAST_IND		b,c					
DL_ESTABLISH_CFM				e			e
DL_RELEASE_IND				b			
CC_STATUS_CFM						b,c	
LCE.04							b
TEMPORARY_ID_ASSIGN_REJECT					e		
DL_ENCRYPT_REQ					e		
DL_ENCRYPT_CFM					e		
DL_DATA_IND					e		
ACCESS_RIGHTS_REQUEST					e		
ACCESS_RIGHTS_TERM_ACCEPT					e		
ACCESS_RIGHTS_TERM_REJECT					e		
CIPHER_REJECT					e		
CIPHER_SUGGEST					e		
IDENTITY_REPLY					e		
LOCATE_REQUEST					e		
TEMPORARY_ID_ASSIGN_ACK					e		

(a)



(b)

[3] Esta vía de comunicación no se ha incluido en el tipo base porque tiene direccionalidad diferente según sea la Terminación Fija o la Terminación Portátil.

SPICUM servicio de publicaciones

Figura K.8: (a) Transiciones posibles y (b) Diagrama de estados para el proceso P_LCE.

K.2.4 Entidad de Gestión de los Niveles Inferiores

La Entidad de Gestión de los Niveles Inferiores (LLME) es la encargada de negociar el establecimiento y la modificación de los servicios solicitados por los usuarios, gestionar los recursos, coordinar los procedimientos de movilidad y cifrado, y dar soporte a los traspasos externos. También almacena y actualiza los parámetros del sistema (IPEI, IPUI, ARI, PARK, …).

El comportamiento se ha modelado de forma que la secuencia de acciones antes de llegar al primer estado son redefinidas en los procesos especializados para las Terminaciones Fija y Portátil. El estado en que se encuentre esta entidad (`DISCONNECTED` o `CONNECTED`) depende de la situación del plano de usuario. La consulta de parámetros se realiza mediante variables exportadas, para reducir el número de señales intercambiadas. Al inicializarse la entidad se notifican los parámetros configurados al Nivel de Enlace mediante la primitiva `LLME_NWK_DLC_IND`.

K.2.4.1 Tipo Base

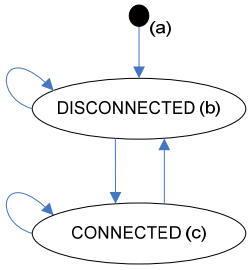
El tipo de proceso base (`LLME`) incluido en el paquete `Comun` define las puertas de comunicación y un conjunto de procedimientos comunes a ambas Terminaciones (Tabla K.6). En esta entidad se ha definido también la respuesta a diversas señales comunes.

Tabla K.6: Procedimientos definidas en el tipo de proceso LLME.

`Conectar`	`Formar<PDU>`

	Estados		
Señales	a	b	c
-	b		
LLME_CONNECT_REQ		b,c	c
LLME_CONFIRMATION_REQ		b,c	c
LLME_DISCONNECT_REQ			b

(a)



(b)

Figura K.9: (a) Transiciones posibles y (b) Diagrama de estados para el proceso LCE.

El diagrama de estados de este proceso (`LLME`) y las transiciones posibles en cada uno de ellos para las señales válidas se muestran en la Figura K.9. El comportamiento de cada estado es el siguiente:

- `DISCONNECTED`: Estado inicial en el que se gestionan las peticiones de establecimiento del Plano U.
- `CONNECTED`: Estado en el que hay establecido un enlace del Plano U.

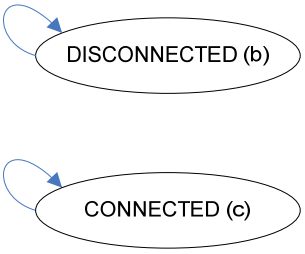
SPICUM servicio de publicaciones

K.2.4.2 Terminación Fija

Este proceso redefine el tipo base añadiendo el procesamiento de varias señales. El diagrama de estados de este proceso (F_LLME) y las transiciones posibles en cada uno de ellos para las señales válidas se muestran en la Figura K.10. El comportamiento de cada estado es equivalente a los del tipo base. También se definen procedimientos específicos para la Terminación Fija.

	Estados	
Señales	b	c
LLME_SETUP_START_IND	b	c
LLME_SETUP_END_IND	b	c
LLME_NOTIFY_IND	b	c

(a)



(b)

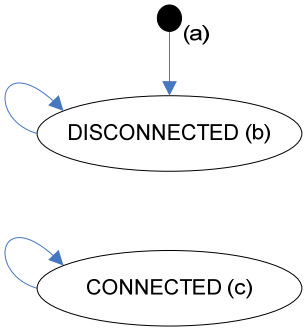
Figura K.10: (a) Transiciones posibles y (b) Diagrama de estados para el proceso F_LLME.

K.2.4.3 Terminación Portátil

Este proceso redefine el tipo base añadiendo el procesamiento de varias señales y modificando la secuencia de acciones en la transición inicial. El diagrama de estados de este proceso (P_LLME) y las transiciones posibles en cada uno de ellos para las señales válidas se muestran en la Figura K.11. El comportamiento de cada estado es equivalente a los del tipo base. También se definen procedimientos específicos para la Terminación Portátil.

	Estados		
Señales	a	b	c
-	b		
LLME_IDENTIFIER_REQ		b,c	c
LLME_ID_IND		b,c	c
LLME_MAC_NWK_IND			b

(a)



(b)

Figura K.11: (a) Transiciones posibles y (b) Diagrama de estados para el proceso P_LLME.

K.3 Pruebas Realizadas

Sobre el Nivel de Red se han realizado dos grupos de Pruebas de Nivel:

- Pruebas con el simulador de SDL: Las pruebas realizadas en el simulador SDL se listan en la Tabla K.7.

SPICUM servicio de publicaciones

Tabla K.7: Lista de Pruebas de Nivel para el Nivel de Red.

Prueba	Objetivo
N01	Establecimiento de conexión iniciado por FT.
N02	Transacción de almacenamiento de la clave de cifrado.
N03	Transacción de actualización del registro de localización.
N04	Pérdida anormal del enlace.
N05	Establecimiento de llamada iniciado por PT.
N06	Autentificación de la Terminación Portátil.
N07	Autentificación de la Terminación Fija.
N08	Identificación de la Terminación Portátil.
N09	Obtención de los derechos de acceso.
N10	Terminación de los derechos de acceso.
N11	Transacción de almacenamiento de la clave de cifrado.
N12	Conmutación del estado de cifrado iniciado por FT.
N13	Conmutación del estado de cifrado iniciado por PT.

- Pruebas utilizando los Juegos de Pruebas Abstractas. Se han realizado de forma similar a como se explica en el Capítulo 8, Sección 8.5.2.3.1; la arquitectura utilizada es la mostrada en la Figura K.12. Los bloques de esta arquitectura se explican también en la sección indicada. El bloque `EMU_IWU_PT` contiene dos procesos que se encargan de interaccionar con `CC` y con `MM`, respectivamente; también ofrece un conjunto de comandos para que el operador pueda controlar la ejecución.

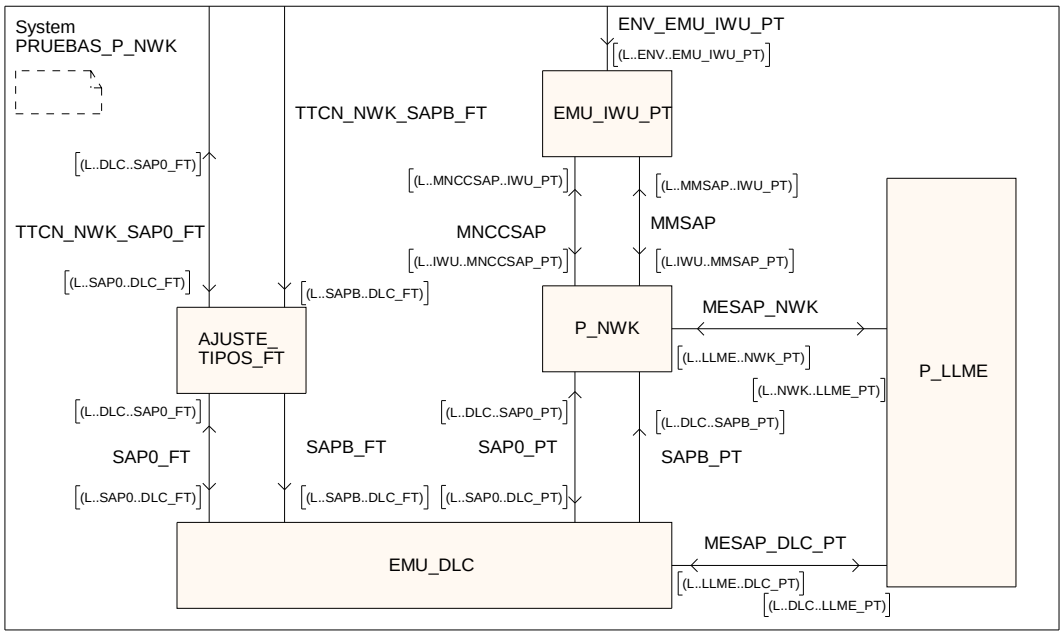


Figura K.12: Arquitectura del sistema SDL empleado para las Pruebas del Nivel NWK de la Terminación Portátil.

SPICUM servicio de publicaciones

Apéndice

L

APÉNDICE L: COMANDOS ABSTRACTOS PARA CONTROL DE LA INSTRUMENTACIÓN

En este apéndice se recogen los comandos, y los parámetros asociados a cada uno, que se han empleado para el control de la Instrumentación en los Sistemas de Pruebas radio de Bluetooth (Tabla L.1) y UMTS (Tabla L.2). Los comandos se muestran en la primera columna, y los parámetros en la tercera. Las columnas segunda y cuarta indican el valor utilizado en los Juegos de Pruebas. Tanto los comandos como los parámetros se han declarado de tipo `IA5String`.

Tabla L.1: Comandos, y parámetros asociados, empleados en Bluetooth para el control de la Instrumentación.

Comando	Valor	Parámetro	Valor
		TSC_par_no_parametro	""
TSC_com_ActMkr1	"ActMarker1"		
TSC_com_ActMkr2	"ActMarker2"		
TSC_com_ActMkr3	"ActMarker3"		
TSC_com_ActMkr4	"ActMarker4"		
TSC_com_ampI	"AMP_I"		
TSC_com_ampQ	"AMP_Q"		
TSC_com_ARM	"Arm"		
TSC_com_BER	"BER"	TSC_par_ON	"ON"
		TSC_par_OFF	"OFF"
TSC_com_bt	"BT"		
TSC_com_center	"Center"	TSC_par_ftx_lowest_79	"2402000000"
		TSC_par_ftx_mid_79	"2441000000"
		TSC_par_ftx_highest_79	"2480000000"
		TSC_par_ftx_lowest_23	"2454000000"
		TSC_par_ftx_mid_23	"2465000000"
		TSC_par_ftx_highest_23	"2476000000"
TSC_com_data	"TipoBER"	TSC_par_PRBS9	"PRBS9"
		TSC_par_PRBS15	"PRBS15"
TSC_com_Delay_Marker1	"DelMarker1"		
TSC_com_Delay_Marker2	"DelMarker2"		
TSC_com_Delay_Marker3	"DelMarker3"		
TSC_com_Delay_Marker4	"DelMarker4"		
TSC_com_DEn	"DataEnable"	TSC_par_Alto	"Alto"
		TSC_par_Bajo	"Bajo"
TSC_com_detector	"Detector"	TSC_par_detector_autopeak	"Autopeak"
		TSC_par_detector_positive	"Positive"
		TSC_par_detector_average	"Average"
TSC_com_findburst	"Findburst"	TSC_par_findburst_on	"ON"
		TSC_par_findburst_off	"OFF"
TSC_com_freq	"Freq"		
TSC_com_frequency	"Frequency"	TSC_par_frequency_on	"ON"
		TSC_par_frequency_off	"OFF"
TSC_com_fskrefdesv	"FSKREFdesv"		

SPICUM servicio de publicaciones

Comando	Valor	Parámetro	Valor
TSC_com_gapsweep	"GapSweep"	TSC_par_gapsweep_on	"ON"
		TSC_par_gapsweep_off	"OFF"
TSC_com_gapsweeppretrig	"GapSweepPreTrig"		
TSC_com_get_BER	"Result"		
TSC_com_get_freq	"Freq?"		
TSC_com_get_peakpower	"PeakPower?"		
TSC_com_IFBw	"IFBw"		
TSC_com_init	"Init"		
TSC_com_Level	"Level"		
TSC_com_Load	"Load"		
TSC_com_Marker	"Marker"		
TSC_com_measfilt	"Measfilter"	TSC_par_measfilt_on	"ON"
		TSC_par_measfilt_off	"OFF"
TSC_com_measure	"Measure"	TSC_par_Auto	"Auto"
		TSC_par_Single	"Single"
TSC_com_memsize	"MemSize"		
TSC_com_mode	"Mode"	TSC_par_mode_maxhold	"Maxhold"
		TSC_par_mode_average	"Average"
TSC_com_modeinstr	"ModeInstr"	TSC_par_vanalyzer	"VectorAnalyzer"
TSC_com_modfsk	"MODFSK"		
TSC_com_ndbdown	"NdBDown"		
TSC_com_NoBits	"Numbits"		
TSC_com_NoError	"NumError"		
TSC_com_norm	"Normalize"	TSC_par_norm_on	"ON"
		TSC_par_norm_off	"OFF"
TSC_com_parar	"Parar"		
TSC_com_polaridad	"Polarity"	TSC_par_Norm	"Normal"
		TSC_par_Inv	"Inverted"
TSC_com_peakpower	"PeakPower"		
TSC_com_ppsymbol	"Ppsymbol"		
TSC_com_rbw	"RBW"		
TSC_com_refilter	"Refilter"	TSC_par_gaussiano	"Gaussian"
TSC_com_reflevel	"RefLevel"		
TSC_com_reset	"Reset"		
TSC_com_restart	"Restart"	TSC_par_int	"Internal"
		TSC_par_ext	"External"
TSC_com_resultlength	"ResultLength"		
TSC_com_RFOut	"SalidaRF"		
TSC_com_Salida_Dig	"ACTDIG"		
TSC_com_SalidaI	"SalidaI"		
TSC_com_SalidaQ	"SalidaQ"	TSC_par_Fix	"Fix"
		TSC_par_Var	"Var"
		TSC_par_Inver	"Inv"
TSC_com_span	"Span"	TSC_par_span_zero	"0"
		TSC_par_span_48M	"48000000"
TSC_com_start	"Start"		
TSC_com_stop	"Stop"		
TSC_com_suelo	"Floor"		
TSC_com_sweep	"Sweep"	TSC_par_cont	"Continuous"
		TSC_par_single	"Single"
		TSC_par_2400s	"2400"
TSC_com_sweepcount	"SweepCount"		
TSC_com_sweeptime	"SweepTime"		
TSC_com_symbolrate	"SymbolRate"		
TSC_com_TRIG	"Trig"		
TSC_com_trigger	"Trigger"	TSC_par_trigger_video	"Video"
		TSC_par_trigger_freerun	"FreeRun"
TSC_com_triggerlevel	"TriggerLevel"		
TSC_com_Unit	"Unit"	TSC_par_Perc	"Percentil"
		TSC_par_ppm	"PPM"
TSC_com_vbw	"VBW"		
TSC_com_Vector	"Vector"		
TSC_com_yperdiv	"YperDivision"		

SPICUM servicio de publicaciones

Tabla L.2: Comandos, y parámetros asociados, empleados en UMTS para el control de la Instrumentación.

Comando	Valor	Parámetro	Valor
		TSC_par_no_parametro	""
TSC_com_center	"Center"	TSC_par_ftx_low	"1922600000"
		TSC_par_ftx_mid	"1950000000"
		TSC_par_ftx_high	"1977400000"
TSC_com_detector	"Detector"	TSC_par_detector_autopeak	"Autopeak"
		TSC_par_detector_positive	"Positive"
		TSC_par_detector_average	"Average"
		TSC_par_detector_sample	"Sample"
		TSC_par_detector_rms	"Rms"
TSC_com_DigStd	"DigitalStandards"	TSC_par_W3GPR	"W3GPPR"
		TSC_par_W3GPF	"W3GPPF"
TSC_com_freq	"Freq"		
TSC_com_frequency	"Frequency"		
TSC_com_gapsweeppretrig	"GapSweepPreTrig"		
TSC_com_gapsweep	"GapSweep"	TSC_par_gapsweep_on	"ON"
		TSC_par_gap	"-0.00001"
		TSC_par_gap25u	"-0.000025"
TSC_com_get_trace	"Trace?"	TSC_par_trace	"trace1"
		TSC_par_trace2	"trace2"
		TSC_par_trace4	"trace4"
		TSC_par_start	"1"
		TSC_par_stop	"0"
TSC_com_IFBw	"IFBw"	TSC_par_1kHz	"1000"
		TSC_par_10kHz	"10000"
		TSC_par_30kHz	"30000"
		TSC_par_50kHz	"50000"
		TSC_par_100kHz	"100000"
		TSC_par_300kHz	"300000"
		TSC_par_1MHz	"1000000"
		TSC_par_5MHz	"5000000"
		TSC_par_10MHz	"10000000"
		TSC_par_25MHz	"25000000"
TSC_com_Magnitude	"Magnitude"		
TSC_com_markerX	"MarkerX"	TSC_par_offset_inicial	"2515000"
		TSC_par_offset_final	"3485000"
		TSC_par_1922M	"1922600000"
TSC_com_MeasResult1	"MeasResult1"	TSC_par_MeasSignal	"MeasSignal"
		TSC_par_RefSignal	"RefSignal"
		TSC_par_ErrorSignal	"ErrorSignal"
TSC_com_MeasResult2	"MeasResult2"	TSC_par_MeasSignal	"MeasSignal"
		TSC_par_RefSignal	"RefSignal"
		TSC_par_ErrorSignal	"ErrorSignal"
TSC_com_mode	"Mode"	TSC_par_mode_maxhold	"Maxhold"
		TSC_par_mode_average	"Average"
		TSC_par_mode_viewtrace	"ViewTrace"
		TSC_par_mode_clearwrite	"Clear/Write"
TSC_com_mode_instr	"ModeInstr"		
TSC_com_mode2	"Mode2"		
TSC_com_ndbdown	"NdBDown"		
TSC_com_PeakPower	"PeakPower?"		
TSC_com_Phase	"Phase"		
TSC_com_PolarIQVector	"PolarIQVector"		
TSC_com_rbw	"RBW"	TSC_par_rbw_30k	"30000"
TSC_com_RealImag	"RealImag"	TSC_par_EVM	"EVM"
		TSC_par_DatosEVM	"DatosEVM"
TSC_com_reflevel	"RefLevel"	TSC_par_ref_0	"0.000000e+00"
		TSC_par_ref_17	"-17.19118"
TSC_com_reset	"Reset"		
TSC_com_ResultLength	"ResultLength"	TSC_par_Length	"250"
TSC_com_ScaleLineal	"Lineal"		
TSC_com_ScaleLog	"Log"		
TSC_com_ScaleUnitY	"ScaleUnitY"		
TSC_com_Signal1	"Signal1"		
TSC_com_Signal2	"Signal2"		
TSC_com_slope	"Slope"	TSC_par_positive	"Positive"
		TSC_par_negative	"Negative"
TSC_com_span	"Span"	TSC_par_span_zero	"0"
		TSC_par_span_10M	"10000000"
		TSC_par_span_2500k	"2500000"
TSC_com_start	"Start"	TSC_par_start_1925M	"1925100000"

SPICUM servicio de publicaciones

Comando	Valor	Parámetro	Valor
TSC_com_stop	"Stop"		
TSC_com_sweep	"Sweep"	TSC_par_continuous_sweep	"Continuous"
		TSC_par_single_sweep	"Single"
		TSC_par_sweep_preamb	"0.00125"
TSC_com_sweep_imm	"SweepImm"	TSC_par_continuous_sweep	"Continuous"
		TSC_par_single_sweep	"Single"
		TSC_par_sweep_preamb	"0.00125"
TSC_com_sweepcount	"SweepCount"		
TSC_com_sweeptime	"SweepTime"	TSC_par_timeslot	"0.000667"
		TSC_par_200ms	"0.2"
		TSC_par_10ms	"0.01"
		TSC_par_20s	"20"
TSC_com_trigger	"Trigger"	TSC_par_trigger_video	"Video"
		TSC_par_trigger_freerun	"FreeRun"
		TSC_par_trigger_level10	"10"
		TSC_par_trigger_level50	"50"
TSC_com_triggerlevel	"TriggerLevel"		
TSC_com_vbw	"VBW"	TSC_par_vbw_30k	"30000"
TSC_com_vector_analyzer	"VectorAnalyzer"		
TSC_com_YperDivision	"YperDivision"	TSC_par_YperDivision	"20"
		TSC_par_Step_Size0	"0"
		TSC_par_Step_Size1	"1"
		TSC_par_Step_Size2	"2"

SPICUM servicio de publicaciones